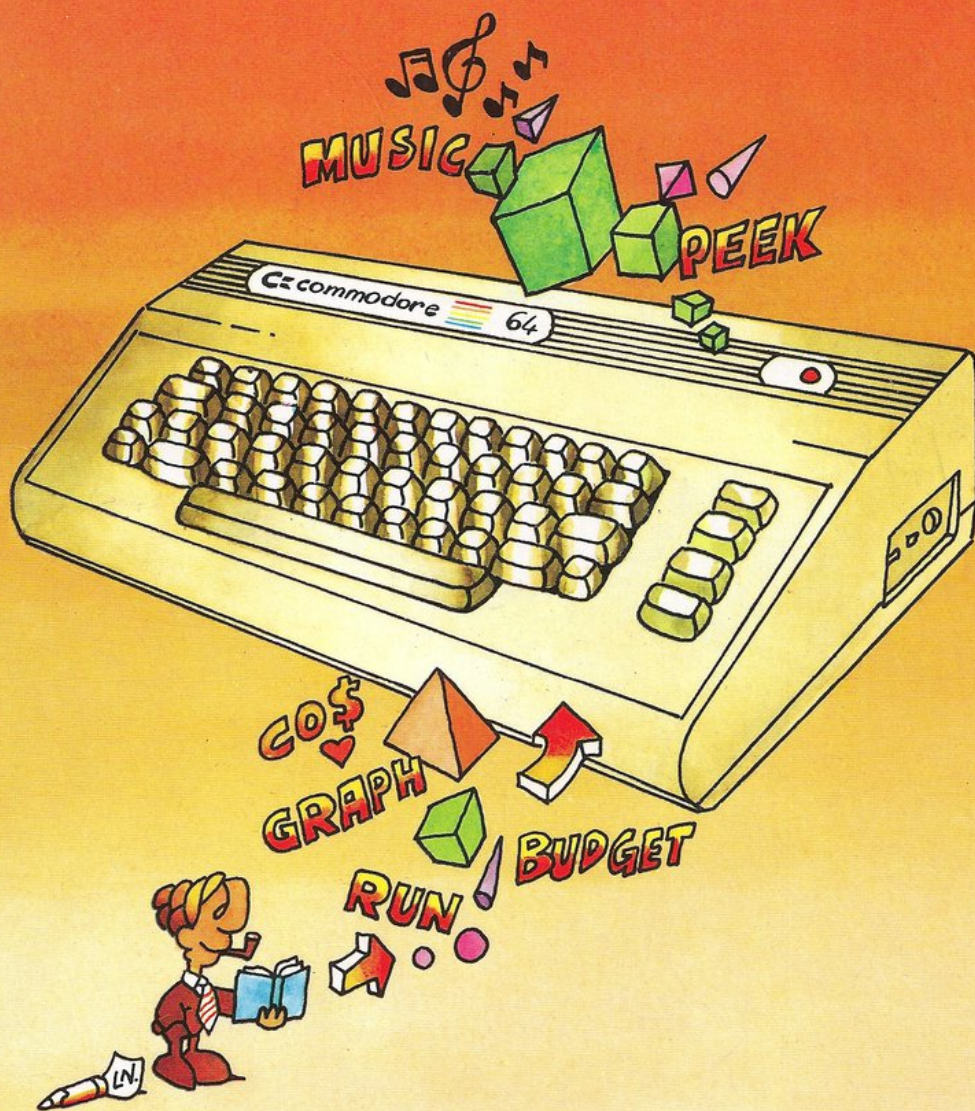
DAVID LAWRENCE

IL MEGLIO DAL COMMODORE 64

programmi efficaci e rapidi
per un ampio ventaglio di applicazioni

Edizioni Elettroniche Serie LIBRI

MUSIC
PEEK
commodore 64
CO$
GRAPH
BUDGET
RUN

I libri delle Edizioni Elettroniche VIFI-Mondadori si rivolgono al pubblico degli utenti, attuali o potenziali, del personal computer e, in genere, a quanti sono interessati agli sviluppi, alle idee e alle tecnologie della cultura informatica e della società dell'informazione.

Volumi pubblicati:

Giulio Banfi
FACILE COME IL BASIC
guida al linguaggio
di programmazione più diffuso
sui personal computer
(5ª edizione)

Ornella Bisio
IL BASIC PER IMPARARE IL BASIC
programmi utili e divertenti
per il personal computer

R. Bradbeer P. De Bono P. Laurie
CAPIRE IL COMPUTER
come sono fatti e che cosa
possono fare gli elaboratori

Aldo Cavalcoli
SCEGLIERE IL PERSONAL COMPUTER
(3ª edizione)

Alberto Cazziol Roberto Galimberti
Marco Maiocchi Roberto Polillo
UN PROGRAMMA CHIAMATO
DOSSIER®
applicazioni di dossier elettronici
per la gestione del lavoro d'ufficio

Vincenzo De Rosso
COME SI PROGRAMMANO
I COMPUTER
(4ª edizione)

Richard E. Pattis
PROGRAMMARE CON KAREL IL ROBOT
l'informatica con carta e matita

Horacio C. Reggini
LOGO: ALI PER LA MENTE
il linguaggio di programmazione
ideato per l'educazione e il gioco creativo

Fausto Servello
CHE COS'È LA TELEMATICA
le nuove tecnologie
della società dell'informazione
(2ª edizione)

Ian Stewart Robin Jones
IL PIACERE DI PROGRAMMARE
CON LO ZX SPECTRUM
dal BASIC alla grafica e alla musica
con uno dei più popolari home computer

Mario Trovato
IL CALCOLATORE TASCABILE
NELLA SCUOLA
un'introduzione all'informatica
(2ª edizione)

Fuori collana:

Luca Novelli
IL MIO PRIMO LIBRO SUI COMPUTER
(2ª edizione)

Luca Novelli
IL MIO PRIMO LIBRO DI BASIC

DAVID LAWRENCE

IL MEGLIO DAL COMMODORE 64

programmi efficaci e rapidi per un ampio ventaglio di applicazioni

ARNOLDO MONDADORI EDITORE

Indice

Presentazione

Questo libro vuole essere un contributo per colmare quella che mi sembra essere una vistosa lacuna nelle collezioni di libri per possessori di microcomputer: l'assenza di lavori intesi a soddisfare quello che ritengo essere il sogno di ogni possessore di un micro, quello cioè che la nuova macchina non diventi semplicemente un giocattolo, e neanche un'istruttiva introduzione all'era del silicio, ma uno strumento di lavoro, in grado di assumersi ogni tipo di compito, e di dischiudere ogni potenzialità. Per contro, la maggior parte dei libri, o consiste di banalità, oppure presuppone un desiderio — e forse anche una capacità — di sperimentare maggiore di quello che sia lecito attendersi.
Desideravo scrivere un libro basato su una collaudata collezione di programmi che potesse valere la pena di possedere — programmi in grado di trattare argomenti come l'archiviazione di dati, la finanza, la grafica, la musica, la gestione famigliare e l'istruzione. La discussione delle tecniche di programmazione sarebbe naturalmente sorta dai programmi stessi, piuttosto che come parte di una serie di "cose da imparare". Spero che il libro che ne è risultato venga considerato utile, non solo come mezzo d'apprendimento di nuove tecniche di programmazione, ma anche come una vera e propria collezione di programmi. Questi sono stati tutti provati autonomamente da un assistente per cercarne gli errori, e possono offrire un campo di applicazioni che avrebbe potuto aprirsi solo a chi fosse disposto a comperare costosi software in commercio, oppure già in grado di scrivere complessi programmi per soddisfare le proprie esigenze.
In questo libro troverete, oltre ai programmi, anche le loro parti — cosa non banale come sembrerebbe, dal momento che i programmi sono scritti in forma "modulare". Ciò significa che sono costituiti da unità funzionali chiaramente identificabili che, una volta comprese, si possono estrarre e reimpiegare secondo necessità. Ogni modulo, dove si riferisca a qualcosa di nuovo, viene completamente commentato, e vengono fornite istruzioni per provare i programmi, man mano che i moduli vengono inseriti.

Benché in alcuni paragrafi vengano affrontati anche argomenti di carattere generale, questo non è un libro da leggere, ma da usare. L'importanza dei vari commenti e consigli si chiarirà solo quando avrete lasciato da parte i tentennamenti per iniziare il lavoro di inserire in macchina quelli che, inizialmente, sembrano programmi così lunghi e complessi da essere scoraggianti. In questa sede, l'approccio modulare aiuterà ad evitare che i programmi diventino inestricabili grovigli di errori; dunque, provate i singoli moduli come viene suggerito, particolarmente agli inizi.

Alla fine, tuttavia, il successo del libro dipenderà dal fatto di essere o meno riuscito a farvi apprezzare appieno il vostro 64. Scrivere un libro come questo è un lavoro pesante, ciò nondimeno ho molto apprezzato la chiarezza che il 64 conferisce ai programmi, che su macchine meno potenti avrebbero potuto essere di gran lunga meno interessanti.

Nel fare uso di questi programmi non vi sarà necessario un lavoro altrettanto faticoso — ma il risultato finale sarà entusiasmante in egual misura.

Infine, non è possibile concludere un'introduzione a un libro come questo senza esprimere un profondo ringraziamento alla Commodore, per tutte le agevolazioni rese disponibili, e un ringraziamento non meno profondo a Steve Beats, della sede inglese della Commodore, per la pazienza dimostrata nel rispondere alle infinite domande relative al 64.

Note ai programmi

Numerose funzioni del Commodore 64, come su altre macchine Commodore, sono dettate da ''caratteri di controllo'', contenuti in normali stringhe, e vengono eseguite allorquando la stringa viene stampata. Solitamente i caratteri di controllo si possono riconoscere per il fatto di essere caratteri inversi (nella posizione ove si trova il carattere, i colori dello sfondo e del simbolo sono scambiati).
Le funzioni poste sotto il controllo di tali caratteri comprendono la posizione del cursore, il colore di stampa, l'attivazione e l'esclusione dell'inversione (RVS), il riposizionamento del cursore in alto a sinistra (HOME) e la cancellazione dello schermo (CLR).
La tabella seguente mostra i caratteri di controllo come appaiono nei programmi di questo libro:

NERO

BIANCO

ROSSO

CIANO

PORPORA

VERDE

BLU

GIALLO

ARANCIO

MARRONE

ROSSO CHIARO

GRIGIO 1

GRIGIO 2

VERDE CHIARO

AZZURRO

GRIGIO 3

RVS ON

RVS OFF

CURSORE SU

CURS. GIU'

CURS. DESTRA

CURS. SINISTRA

HOME

CLR

1. Programmi utili e compatti

I programmi contenuti in questo libro sono pensati per essere posti al lavoro su una quantità di applicazioni di una certa importanza. Dato che molte delle applicazioni sono complesse, tali risultano anche molti dei programmi. Ciò non significa che dei programmi utili non si possano comprimere in uno spazio limitato. A scopo introduttivo verso l'approccio adottato, questo capitolo presenta tre programmi relativamente brevi che sono tutt'altro che giochi.

1.1 Orologio

Questo programma fornisce una piacevole introduzione ad alcune delle possibilità del 64, è facile da introdurre, divertente da lasciare girare sul TV di casa, e fa buon uso della versatilità di operazione sullo schermo e sulle stringhe che è propria del 64.

Il programma è esattamente ciò che dice di essere, un orologio, ma quando viene mandato in esecuzione non vedrete apparire un cerchio e delle lancette. L'orologio "64" utilizza due linee che scandiscono lo schermo, da sinistra a destra per i minuti e verso il basso per le ore, dividendo lo schermo in aree diversamente colorate. Tutto ciò è possibile solo dal momento che il 64 possiede una versatile funzione di conteggio del tempo che si può inizializzare e leggere in modo diretto dall'interno del programma.

Orologio: lista delle variabili

IC	Indirizzo d'inizio della memoria dei colori
DT$	Adattamento formattato di TI$
H	Valore dell'ora in unità dello schermo
M	Valore del minuto in unità dello schermo
M1$ M2$	Stringhe di due colori che mostrano i valori dell'ora e del minuto
IS	Indirizzo d'inizio dello schermo
TI$	Variabile di sistema contenente il tempo scandito dall'orologio interno

MODULO 1.1.1

```
11000 REM*****************************
11010 REM INIZIALIZZ. ORE E SCHERMO
11020 REM*****************************
11030 POKE 53280,0:POKE 53281,1
11040 INPUT "INTRODURRE L' ORA (01-12
): ";H$
11050 INPUT "INTRODURRE IL MINUTO ESAT
TO (00-59): ";M$
11060 TI$=H$+M$+"00"
11070 PRINT"     5 10 15 20 25 30 35 4
0 45 50 55 60  "
11080 IS=1024:IC=55296:FOR I=0 TO 24
11090 POKE IC+40*I,0:POKE IS+40*I,160
11100 POKE IC+40*I+1,0:POKE IS+40*I+1,16
0
11110 POKE IC+40*I+38,0:POKE IS+40*I+38,
160
11120 POKE IC+40*I+39,0:POKE IS+40*I+39,
160
11130 NEXT I
11140 PRINT"":FOR I=1 TO 11:PRINT"";MI
D$(STR$(I),2):PRINT:NEXT I
11150 PRINT"";MID$(STR$(I),2);
```

Questo modulo permette all'utente di introdurre le ore e i minuti (orologio con formato di 12 ore), inizializza il contatore e mostra quindi il quadrante dell'crologio.

Commenti

Linea 11030: due utili locazioni di memoria: 53280 ridefinisce il colore del bordo intorno allo schermo, 53281 impone il colore di fondo dello schermo. Entrambe si possono riportare istantaneamente alle condizioni iniziali nel corso di un programma. In questo caso il bordo appare nero e lo schermo bianco.

Linee 11040-11060: le ore e i minuti sono inseriti nel formato di due cifre. Vengono sommati tra di loro con l'aggiunta di 00 per i secondi. Si dice al sistema che questo è TI$ e immediatamente il sistema riposiziona l'orologio interno per contare da quell'ora.

Linea 11070: viene pulito lo schermo, il colore dei caratteri posto a nero e attivata l'inversione, quindi lungo il lato superiore dello schermo vengono stampati i numeri.

Linee 11080-11130: ai lati dello schermo vengono ora posti i bordini neri dell'area dell'orologio. Quando si stampa all'estremità dello schermo, è spesso più conveniente porre i caratteri sullo schermo tramite una POKE, dal momento che ciò evita che la posizione di stampa salti alla linea successiva. Per portare a termine una POKE sullo schermo con esito favorevole, bisogna trattare due locazioni: una entro la memoria di schermo stessa (indirizzi 1024-2023) e la seconda nella memoria colore (55296-56295). Tutto quello che fa questo

ciclo è di attribuire il codice carattere 160 (spazio inverso) ai primi e agli ultimi due caratteri delle 25 linee dello schermo e il codice 0 alle corrispondenti locazioni della memoria colore, cosa che annerisce quelle posizioni.

Linea 11140: si posiziona il cursore in alto a sinistra e lungo il lato sinistro dello schermo vengono stampate le ore. L'ultimo valore viene stampato separatamente seguito da un punto e virgola, in modo da non far scorrere verso l'alto lo schermo, dal momento che si trova sull'ultima linea.

Collaudo del modulo 1.1.1

Inserite la linea temporanea 11160 GOTO 11160 e mandate in esecuzione il modulo. Vi si richiederà d'inserire le ore e i minuti, appariranno quindi sullo schermo i bordi del quadrante dell'orologio. La linea temporanea fa sì che lo schermo non scorra verso l'alto per stampare READY, una volta terminato il modulo.

MODULO 1.1.2

```
12000 REM********************************
12010 REM CALCOLA E VISUALIZZA ORA
12020 REM********************************
12030 M=INT((VAL(MID$(TI$,3,2))+0.8)*3/5
)
12040 H=2*VAL(MID$(TI$,1,2))
12050 IF M>=30 THEN LET H=H+1
12060 IF H>=24 THEN LET H=H-24
12070 IF M=0 THEN LET M=1
12080 M1$="▮▮▮▮
          "
12090 LET M1$=LEFT$(M1$,M+4)+"▮"+RIGHT$(
M1$,36-M)
12100 M2$="▮▮▮▮
          "
12110 LET M2$=LEFT$(M2$,M+4)+"▮"+RIGHT$(
M2$,36-M)
12120 PRINT"▮▮";
12130 IF H>0 THEN FOR I=1 TO H:PRINTM1$:
NEXT I
12140 IF H<23 THEN FOR I=H+1 TO 23:PRINT
M2$:NEXT I
12150 PRINTM2$;
12160 PRINT"▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮
▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮";
12170 DT$=LEFT$(TI$,2)+"   "+MID$(TI$,3,2
)+"   "+RIGHT$(TI$,2)
12180 FOR I=1 TO LEN(DT$):PRINT MID$(DT$
,I,1);"▮▮";:NEXT I
12190 GOTO 12030
```

Questo modulo ricava dall'orologio interno i valori necessari per creare il display e mostra l'ora in due forme differenti.

Commenti

Linea 12030: una volta tracciati i bordi, sullo schermo ci sono 36 spazi disponibili, dunque il numero dei minuti va diviso per 60/36 (5/3) per ottenere la conversione in unità di schermo.

Linea 12040: sono disponibili 24 linee, così tutto quel che c'è da fare è moltiplicare le ore per 2.

Linea 12070: il programma è progettato per indicare sempre i minuti, così, in coincidenza con l'ora, il valore dei minuti viene incrementato per mostrare un'unità.

Linea 12080: M1$ viene posta uguale a due spostamenti a destra del cursore, più il carattere di controllo per il colore porpora e il carattere di controllo per attivare l'inversione, seguito da 36 spazi. Se stampata, questa stringa produrrebbe una linea in colore porpora.

Linea 12090: M1$ viene ora modificata in modo da contenere i primi quattro caratteri di controllo seguiti da M spazi, quindi un carattere di controllo per il rosso seguito dagli spazi rimanenti. Ciò crea una nuova stringa che cambia colore ad un punto definito dal valore di M.

Linee 12100-12110: lo stesso procedimento si segue con M2$, che inizierà in blu e terminerà in bianco.

Linee 12130-12150: M1$ viene stampata per tante linee quante sono le ore; M2$ viene stampata per le linee rimanenti. Le due stringhe definiscono così un confine tra aree di colore diverso, imposte dal valore di H.

Linea 12160: si riporta il cursore in alto a sinistra e si sposta la posizione di stampa verso il basso di circa un terzo fino alla penultima colonna, con colore impostato sul nero.

Linea 12170: DT$ viene ora definita come TI$ con l'aggiunta di due spazi inseriti tra i valori dell'ora, dei minuti e dei secondi. Nel corso dell'esecuzione, il sistema ha continuato ad aggiornare TI$, che quindi contiene sempre l'ora esatta.

Linea 12180: DT$ viene stampata verticalmente lungo il lato destro dello schermo. La stampa in verticale si ottiene con il metodo di scrivere carattere per carattere, ogni volta spostando il cursore di una posizione indietro e di una verso il basso.

Collaudo del modulo 1.1.2

A questo punto il vostro orologio dovrebbe essere pronto a funzionare, con quattro diversi rettangoli colorati che segnano le linee delle ore e dei minuti. L'ora viene inoltre mostrata numericamente sul lato destro dello schermo.
Detto quello che dovrebbe fare, quasi inevitabilmente ci saranno degli errori in ciò che avete introdotto. E se non in questo, nei programmi successivi. Dal tipo di domande che mi vengono poste, mi sem-

bra che molti possessori di microcomputer incontrino notevoli difficoltà per mettere a punto un metodo di ricerca ed eliminazione degli errori (*debugging*) e con tutta probabilità alcune indicazioni fondamentali possono risultare utili:

1) Fate in modo di utilizzare al meglio ogni tipo di aiuto. Se compare un messaggio d'errore, assicuratevi di tenerlo in giusto conto, prendendo nota del tipo di errore e della linea dove ha avuto luogo.
2) Non tentate di fare eseguire il programma una seconda volta se non ha funzionato la prima. Se dovesse effettivamente funzionare, vi trovereste in una condizione peggiore di quella iniziale, avendo per il momento perso ogni possibilità di scoprire gli errori.
3) Usate il modo diretto (comandi inseriti direttamente dalla tastiera, invece che come linee di programma) per far stampare i valori di tutte le variabili delle linee che appaiano contenere errori. Un valore privo di senso spesso vi darà l'indizio risolutivo su quello che non sta funzionando. Un'incredibile quantità di errori quasi introvabili discende da un semplice errore di scrittura del nome di una variabile, come il battere per esempio un 1 al posto di I.
4) Seguite il programma mentalmente o per iscritto, usando valori semplici, in modo da poter vedere esattamente quello che dovrebbe fare in ogni istante.
5) Non siate avventati nell'introdurre una modifica, finché non siete sicuri che è l'unica che volete apportare. Una volta alterata una linea, tutti i vostri dati spariscono, e con questi ogni possibilità di ulteriori prove senza eseguire nuovamente il programma.
6) Salvate regolarmente il programma, quando scoprite degli errori e/o aggiungete nuove linee. Numerosi errori nei programmi definitivi si possono ascrivere a variazioni apportate nel testo del programma, ma alla fine mai registrate su nastro. Tutti i miei programmi iniziano con le seguenti tre linee:

```
1 GOTO 3
2 SAVE "XXXX": STOP
3 REM
```

Queste fanno sì che sia possibile salvare i programmi tramite il comando GOTO 2 (posto di aver sostituito "XXXX" con il nome del programma). Un beneficio marginale è che posso sempre iniziare un'esecuzione con GOTO 1, invece di dover ricordare il primo numero di linea del programma principale.

Tutti fanno errori nel progettare e battere un programma, la differenza sta solo nel modo in cui si impara a trattare i propri errori con competenza.

Riepilogo

Che questo orologio vi piaccia o meno, è solo una questione di gusti. Personalmente lo trovo abbastanza gradevole. Al di là dell'orologio, tuttavia, le tecniche di frammentazione delle stringhe e di POKE delle memorie di schermo e di colore contenute nel programma, si dimostreranno utili in una varietà di applicazioni. Vale dunque

la pena d'inserire il programma e di assicurarsi di comprenderne appieno il funzionamento.

1.2 Grafica

Se volete farvi un'idea di questo programma, non c'è niente di meglio che gettare uno sguardo alla scatola in cui era imballato il vostro 64. Su questa troverete una colorata rappresentazione tridimensionale a barre: quello che segue è un tentativo di riprodurre il programma che ha generato quella figura.
Dico tentativo, dato che, riproducendo la figura sulla scatola, ho scoperto che i dati forniti per l'elaborazione erano stati attentamente scelti per nascondere le limitazioni che sorgono in caso di barre molto ravvicinate. Dati diversi conducevano i caratteri grafici costituenti le barre a scontrarsi con le barre adiacenti, rendendo il tutto molto meno gradevole del disegno sulla scatola.
Perciò questo programma è un compromesso, in grado di produrre immagini meno intricate, ma funzionante per qualsiasi insieme di dati con risultati sempre ugualmente soddisfacenti: tanto che, quando avrete terminato d'inserirlo, sarà il tipo di programma che vi farà chiamare tutti i parenti per impressionarli con la vostra magia.
Utili e colorate, le figure prodotte troveranno senza dubbio diverse applicazioni. Inoltre il programma fornisce una visione introduttiva su come salvare i dati su nastro per caricarli successivamente.

Grafica: lista delle variabili

CO$	Stringa di tre caratteri utilizzata per stabilire il colore delle diverse barre sul grafico
F$	Stringa di formattazione composta da caratteri per lo spostamento a destra del cursore
F1$, F3$	Stringhe di formattazione composte da caratteri per lo spostamento del cursore verso il basso
F2$	Stringa temporanea ricavata da F$
HH(2,6)	Matrice contenente i dati per il grafico
NB	Numero di banchi uno di fronte all'altro (1-3)
ND	Numero di colonne sull'asse orizzontale (1-6)
NO$	Nome dell'asse orizzontale
NV$(2)	Nomi di ogni singolo banco
TT$	Stringa temporanea usata per formattare la stampa dei nomi dell'asse verticale
UV	Numero rappresentato da ogni unità dell'asse verticale

MODULO 1.2.1

```
11000 REM#****************************
11010 REM INTRODUZIONE DATI
11020 REM*****************************
11030 POKE 53281,15:INPUT "▯▮▯E' NECESSA
RIO CARICARE DAL NASTRO (S/N):";Q$
11040 IF Q$="S" THEN 12420
11050 PRINT"▯▯▮▮▮▮▮▮▮▮▮▮▮▮▮▯▯GRAFICO"
11060 PRINT"▯▮SONO PRESENTI 19 UNITA' VE
RTICALI."
```

```
11070 PRINT"INTRODURRE IL VALORE RAPPR
ESENTATO DA "
11080 INPUT"OGNI UNITA': ";UV
11090 INPUT"NOME DELL' ASSE ORIZZONTAL
E";NO$
11100 PRINT"SI POSSONO AVERE DA UNA A
 SEI COLONNE"
11110 INPUT"QUANTE NE SERVONO: ";ND
11120 PRINT"SI POSSONO AVERE DA UNO A
 TRE BANCHI"
11130 INPUT"QUANTI NE SERVONO: ";NB
11140 FOR I=0 TO NB-1
11150 PRINT"NOME DELL' ASSE VERTICALE"
;I+1;:INPUT NV$(I):NEXT I
11160 DIM HH(2,6)
11170 PRINT"";:FOR I=0 TO NB-1
11180 FOR J=1 TO ND
11190 PRINT"INTRODURRE IL VALORE DELLA
COLONNA ";I+1;J;":";:INPUT T
11200 IF INT(T/UV)>19 THEN PRINT"VALORE
 TROPPO ELEVATO.":GOTO 11190
11210 HH(I,J)=T:NEXT J,I
```

Lo scopo di questo semplicissimo modulo è di permettere l'introduzione dei dati che si useranno per costruire e marcare (*label*) il grafico. Invece di chiedere all'utente il numero massimo e minimo della gamma di valori, per poi calcolare le unità di misura (cosa che può portare a valori molto strani), il programma domanda all'utente di specificare quante unità dei dati introdotti saranno rappresentate da ogni spazio unitario verticale sul grafico. Si assegnano dei nomi all'asse orizzontale e ai banchi di colonne tridimensionali, a cominciare dal fondo. Alla fine vengono richiesti i dati conformemente alla struttura scelta.

Collaudo del modulo 1.2.1

È semplicemente questione di far "girare" il modulo per controllare che sia esatta la sintassi. Non si può ancora disegnare niente.

MODULO 1.2.2

```
12000 REM#******************************
12010 REM DISEGNO GRAFICO
12020 REM*******************************
12030 POKE53281,0:PRINT "
";
12040 F$=""
12060 FOR I=1 TO 4:PRINT F$;"
                       ":F$=F$+"":NEXT
12070 PRINT "______________________
_________":REM LINEA DI 30 COM & P
12080 FOR I=1 TO 19:PRINT"L";
12090 PRINT "
 ┘":NEXT
```

```
12100 PRINT "";:FOR I=1 TO 4:PRINT "";
12110 PRINT "
":NEXT:REM LINEA DI 28 COM & P
12120 F1$="":F$="":CO$=""
12130 PRINT "";
NO$
12140 FOR H=0 TO NB-1:PRINT "";
MID$(F$,1,2*(H+1));MID$(CO$,H+1,1);
12150 TT$=NV$(H)+" *"+STR$(UV):FOR I=1 T
O LEN(TT$)
12160 PRINT MID$(TT$,I,1);"";:NEXT I,H
12170 PRINT "15105
"
12180 F3$=F1$:FOR H=0 TO NB-1:PRINT MID$
(CO$,H+1,1)
12190 F2$=LEFT$(F$,8+4*(ND-1)+H):PRINT F
1$;F2$;
12200 FOR I=ND TO 1 STEP-1:IF INT(HH(H,I
)/UV)=0 THEN 12270
12210 FOR J=1 TO INT(HH(H,I)/UV)+H
12220 IF INT(HH(H,I)/UV)=0 THEN 12270
12230 IF J=1 THEN PRINT "";MID$(CO$,
H+1,1);"|";
12240 IF J>1 THEN PRINT " | ";
12245 REM I CARATTERI SONO " | . . | . . .
. . . . ";
12250 NEXT J
12260 PRINT""
12270 PRINT:F2$=LEFT$(F2$,LEN(F2$)-4):PR
INT F1$;F2$;:NEXT I
12280 F1$=F1$+""
12290 F2$=LEFT$(F$,9+4*(ND-1)):PRINT F3$
;F2$;
12300 NEXT H
12310 FOR I=1 TO ND:PRINT F3$;
12320 F2$=LEFT$(F$,9+4*(I-1)):PRINT F2$;
:FOR J=1 TO NB
12330 IF J>1 OR HH(2,I-1)=0 THEN PRINT "
";
12340 PRINT "";:NEXT J,I
12350 GET A$:IF A$="" THEN 12350
```

Questo è un modulo ''disonesto'', basato non su un qualsiasi chiaro insieme di metodi, ma semplicemente sulle condizioni che, in pratica, ho scoperto debbono essere soddisfatte per completare una piacevole rappresentazione della figura.

Commenti

Linee 12040-12060: si annerisce lo schermo e viene colorata la base marrone su cui posa la figura.

Linee 12070-12110: viene tracciata la griglia che circonda l'area del grafico, verticalmente lungo i lati vengono segnate le unità di misura, mentre orizzontalmente vengono poste delle linee per segnare i cinque livelli unitari.

Linee 12140-12160: queste tre linee determinano una posizione di stampa nell'angolo superiore destro dello schermo, dove stampano, nei colori corrispondenti e in senso verticale, i nomi dei tre banchi. La posizione sullo schermo viene determinata tramite uno spezzone di F$, mentre il colore dipende dalla stampa di uno, diverso per ogni iterazione del ciclo, dei tre caratteri di controllo colore.

Linee 12180-12300: in questa sezione vengono visitati tre cicli. Il ciclo H stabilisce quanti banchi verranno tracciati uno davanti all'altro; viene inoltre utilizzato per estrarre un carattere di controllo del colore da CO$ e per determinare quanti caratteri spostamento verso il basso del cursore verranno stampati, muovendo dunque in conseguenza i banchi in senso verticale. Il ciclo I controlla il numero di colonne da stampare attraverso lo schermo; il ciclo J l'altezza di ogni singolo blocco tridimensionale.

Linea 12190: si calcola, a seconda del banco, la posizione iniziale di stampa in senso orizzontale (vengono tracciati da destra a sinistra) — ogni banco si muove in orizzontale di una posizione, dando un aspetto tridimensionale ai tre banchi.

Linea 12200: non si procede al disegno della colonna se l'elemento interessato nella matrice HH vale zero.

Linee 12230-12240: alla base di ogni colonna viene stampato il supporto inclinato che la fa sembrare appoggiata alla superficie.

Linea 12260: una volta raggiunta l'estremità superiore della colonna, viene aggiunta la cima inclinata.

Linea 12270: per la colonna successiva vengono sottratti quattro caratteri dalla stringa di spostamento a destra del cursore, definendo così la nuova posizione di stampa.

Linea 12280: prima di stampare il banco successivo, si sposta verso il basso la posizione di stampa verticale.

Linee 12310-12340: con la stampa dei banchi, si sono rovinate le basi delle colonne. Ora vengono riempite.

Linea 12350: la figura rimane sullo schermo fino a che non si preme un tasto.

Collaudo del modulo 1.2.2

Ancora una volta, è semplicemente questione di eseguire il programma e controllare l'esattezza della figura risultante. In caso di problemi, la soluzione è di ridurre a uno il numero di banchi, eventualmente anche quello delle colonne, per semplificare l'analisi di ciò che sta andando storto.

Osservate separatamente la funzione di ogni singolo ciclo per decidere quale sia responsabile dell'errore. È questo un modulo sul quale può risultare frustrante eseguire il debugging, così, se ritenete che una modifica possa essere di giovamento, apportatela pure e provate, anche se non siete in grado di stabilire in che modo vi siete distaccati dal listato qui fornitovi — niente è sacrosanto in questo libro, qui o altrove.

MODULO 1.2.3

```
12360 INPUT "E' NECESSARIO SALVARE I
DATI (S/N): ";Q$:IF Q$="N" THEN END
12370 INPUT "POSIZIONARE IL NASTRO, QUI
NDI PREMERE RETURN--";Q$:R$=CHR$(13)
12380 OPEN 1,1,1,"GRAFICO"
12390 PRINT#1,NB;R$;ND;R$;NO$,R$;UV
12400 FOR I=0 TO NB-1:PRINT#1,NV$(I):FOR
 J=0 TO ND:PRINT#1,HH(I,J):NEXT J,I
12410 CLOSE 1:END
12420 INPUT "POSIZIONARE IL NASTRO, Q
UINDI PREMERE RETURN--";Q$:DIM NV$(2):
12421 DIM HH(2,6)
12430 OPEN 1,1,0,"GRAFICO"
12440 INPUT#1,NB,ND,NO$,UV
12450 FOR I=0 TO NB-1:INPUT#1,NV$(I):FOR
 J=0 TO ND:INPUT#1,HH(I,J):NEXT J,I
12460 CLOSE1:GOTO 12000
12470 GOTO 12470
```

Ora che avete definito la vostra figura, invece di perdere i dati e doverli poi reintrodurre, potete immagazzinarli su nastro. Questo modulo ve ne darà la possibilità, e potrete così richiamarli in seguito. Il modulo è progettato in modo da rendere l'immagazzinamento su nastro il più facile possibile, dal momento che vi dà il tempo di posizionare correttamente il nastro, prima d'iniziare il processo di salvataggio o caricamento.

Commenti

Linea 12380: questa linea apre un *file*, un luogo dove vanno posti i dati; in questo caso il luogo d'immagazzinamento sarà il registratore a cassette. I tre numeri rappresentano:
a) il numero del file — qualsiasi istruzione per memorizzare qualcosa in un file deve citare il numero del file;
b) il numero del dispositivo (l'apparecchio fisico che deve ricevere i dati) — dove 1 rappresenta il registratore a cassette;
c) il tipo di file — 1 significa che si tratta di un file in cui porre i dati, piuttosto che uno da cui raccoglierli.
NB, ND, NO$ e UV vengono posti nel file (sul nastro). Notate in questa sede l'uso della variabile R$. Durante la memorizzazione di dati

su nastro, il 64 è un poco schizzinoso sul modo in cui ogni elemento viene separato dal successivo — la semplice inserzione di virgole può condurre a errori durante il successivo caricamento dei dati. R$ è stata definita alla linea 12370 come CHR$ (13), il codice di RETURN, e, inserendola tra ogni elemento, se ne ottiene la corretta separazione.

Linea 12400: le matrici del programma vengono stampate nel file una per una.

Linea 12410: è necessario chiudere (CLOSE) un file quando avete momentaneamente finito di utilizzarlo. Dimenticando questa operazione, otterrete un errore quando cercherete di aprire (OPEN) un file dotato dello stesso numero.

Linea 12420: si tratta della parte del modulo che ricarica i dati nel 64. La sola differenza tra le specifiche dei due tipi di file è che questo è un tipo 0, vale a dire un file dal quale estrarre i dati.

Linee 12440-12450: vengono qui richiamati i dati che erano stati stampati nel file. Il modo più sicuro per costruire la vostra routine di caricamento è quello di modificare i numeri di linea della routine SAVE e trasformare le istruzioni PRINT in INPUT. In questo modo sarete certi che la routine raccoglierà i dati esattamente nello stesso ordine in cui erano stati memorizzati. Se i dati venissero raccolti in una sequenza erronea, non solo non avrebbero senso con il vostro programma, ma potrebbe anche insorgere un errore nella terminazione dell'esecuzione.

Collaudo del modulo 1.2.3

La semplice prova per questo modulo consiste nel verificare la possibilità di introdurre dei dati nel programma, salvarli sul nastro e poi ricaricarli.

Riepilogo

Questo programma è un omaggio alla qualità del set grafico del 64. Una volta inserito, comincerete a comprendere che non è affatto una cosa così difficile, per mezzo di cicli e semplici calcoli, disegnare forme e oggetti apparentemente solidi in posizioni prestabilite dello schermo, e scoprirete che visualizzare in tal modo dati e problemi rappresenta un efficace contributo per la loro comprensione e soluzione.

1.3 Testi

Il programma conclusivo di questo capitolo è un tentativo di fornire alcune delle funzioni più semplici di un *word processor* in un programma relativamente breve e non complicato. Naturalmente, il programma non è paragonabile a un word processor scritto in codice macchina e di fattura professionale, né sarebbe lo strumento che sceglierei per scrivere un libro come questo, non foss'altro perché, allo scopo, alla Commodore inglese hanno avuto la gentilezza di fornirmi una copia del loro eccellente programma Easyscript. Cionondimeno il programma funziona, e l'avrei preferito a una macchina da scrivere in numerose applicazioni, a causa della versatilità che offre

nello stendere e modificare un testo prima di porlo definitivamente sulla carta. Naturalmente, se non possedete una stampante, dovete precipitarvi a comperarne una, per poter avere il massimo dal programma.

Testi: lista delle variabili

A$	Linea di testo da introdurre
CH	Codice del carattere sotto il cursore lampeggiante
FNA(P)	Calcola la posizione in memoria del cursore lampeggiante nel testo da introdurre
FNB(P2)	Calcola la posizione in memoria del cursore di correzione e modifica (*edit*) nel corpo principale del testo
LL	Numero di linee nel corpo principale del testo
P	Posizione del cursore lampeggiante nella linea da introdurre
P2	Posizione del cursore di edit sullo schermo
PL	Numero della linea del cursore di edit nel corpo principale del testo
SP	Numero di spazi disponibili al termine della linea durante la formattazione
SS	Numero della prima linea della parte da visualizzare del testo principale
T$	Ultimo carattere introdotto quando si inserisce una nuova linea di testo
T1$	Carattere introdotto come comando nel modulo Movimento Linea di Edit
TEXT$(500)	Vettore principale per l'archiviazione del testo
TT$	Archiviazione temporanea delle linee da introdurre nel corpo principale del testo
X	Numero delle linee estratte dallo spezzone di testo da inserire nel corpo principale

MODULO 1.3.1

```
11000 REM#******************************
11010 REM INIZIALIZZAZIONE
12000 REM*******************************
12010 PRINT "[CLR]";:DIM TEXT$(500):LL=1:PL=
1
12020 TEXT$(0)="[graphic characters]"
12030 TEXT$(1)="[graphic characters]"
12040 DEF FNA(P)=1024+20*40+P
12050 DEF FNB(P2)=1024+40*P2
12060 GOSUB 14110
```

Questo modulo inizializza le variabili principali e pone dei marcatori d'inizio e fine testo all'interno del vettore principale.

MODULO 1.3.2

```
13000 REM#******************************
13010 REM CORREZIONE LINEA
13020 REM*******************************
```

```
13030 A$=" "
13040 P=0:PRINT "[illegible]"
;A$
13050 CH=PEEK(FNA(P)):POKE 54272+FNA(P),
14:POKE FNA(P),160
13060 FOR TT=1 TO 5:NEXT TT:POKE FNA(P),
CH
13070 GET T$:IF T$="" THEN GOTO 13050
13080 IF T$=CHR$(13) OR (LEN(A$)>80 AND
T$=" ")THEN GOSUB 14000:GOTO 13050
13090 IF T$="↑" THEN GOSUB 15000:POKE FN
B(PL-SS),62:GOTO 13040
13100 IF T$="←" AND P<>0 THEN 13040
13110 IF T$="←" AND P=0 THEN P=LEN(A$)-1
:GOTO 13150
13120 IF P>0 AND T$=CHR$(20) THEN A$=LEF
T$(A$,P-1)+MID$(A$,P+1):P=P-1:GOTO13150
13130 IF T$=CHR$(20) THEN13050
13140 IF T$<>"" AND T$<>"▮"ANDT$<>"▶▮"THE
N A$=LEFT$(A$,P)+T$+MID$(A$,P+1):P=P+1
13150 PRINT "[illegible]";A$:I
F T$="▮" AND P>0 THEN P=P-1
13160 IF T$="▶▮" AND P<LEN(A$)-1 THEN P=P
+1
13170 GOTO 13050
```

Lo scopo di questo modulo è di permettere all'utente di introdurre e modificare fino a due linee di testo sul lato inferiore dello schermo e di correggere le linee in preparazione, per inserirle nel corpo principale del testo tramite un modulo successivo.

Commenti

Linee 13050-13070: la prima routine con cursore lampeggiante incontrata. Basandosi sulla funzione-utente A, queste linee (tramite PEEK) controllano la memoria di schermo nel punto definito dalla variabile P e stabiliscono il codice del carattere che vi si trova allocato. Nella stessa locazione viene forzato (POKE) uno spazio blu inverso, che là viene lasciato per la durata di un breve ciclo di temporizzazione, per poi essere sostituito dal carattere originario. Se non si è battuto niente dalla tastiera, come indicato dall'enunciato GET, il processo viene ripetuto.

Linea 13080: premendo RETURN, questa linea inserisce la riga corrente nel corpo principale del testo — è necessario un modulo successivo. La linea viene inserita automaticamente anche nel caso ecceda la lunghezza di due linee dello schermo e questo potrebbe eliminarne i caratteri finali.

Linea 13090: l'azionamento del tasto ↑ permette di accedere a un modulo successivo che agisce sul corpo principale del testo.

Linee 13100-13110: premendo il tasto con il simbolo della freccia a sinistra (all'estremità superiore sinistra della tastiera), il cursore si sposta all'inizio o alla fine della linea, a seconda della sua posizione corrente.

Linee 13120-13130: posto che il cursore non si trovi all'inizio della linea, l'azionamento di DELETE elimina il carattere immediatamente precedente il cursore. Notate che, usando GET, i tasti di controllo come DELETE non producono alcun effetto a meno che non vengano stampati, cosicché, se non li stampiamo, possiamo ridefinire la loro funzione.

Linea 13140: se il carattere premuto non è una freccia di movimento del cursore, allora si considera un carattere da stampare, e viene aggiunto alla stringa nella posizione del cursore. Se il cursore si trova nel mezzo della stringa, il carattere viene inserito in aggiunta — non sostituisce cioè il carattere sotto il cursore.

Linee 13150-13160: la stringa viene nuovamente stampata nella sua forma modificata. Se l'ingresso era una freccia di movimento del cursore, la posizione di quest'ultimo viene modificata in conseguenza.

Collaudo del modulo 1.3.2

Inserendo provvisoriamente RETURN alla linea 14000 dovreste poter ora scrivere e correggere un testo sul lato inferiore dello schermo.

MODULO 1.3.3

```
14000 REM********************************
14010 REM INSERZIONE LINEA
14020 REM********************************
14030 X=0
14040 IF LEN(A$)<40 THEN TT$(X)=LEFT$(A$
,LEN(A$)-1):A$="":GOTO 14070
14050 FOR I=40 TO 1 STEP-1:IF MID$(A$,I,
1)<>" "THEN NEXT I:I=40
14060 TT$(X)=LEFT$(A$,I-1):A$=MID$(A$,I+
1)
14070 X=X+1:IF A$<>"" AND A$<>" " THEN G
OTO 14040
14080 FOR I=LL+X TO PL+X STEP-1:TEXT$(I)
=TEXT$(I-X):NEXT I
14090 FOR I=0 TO X-1:TEXT$(PL+I)=TT$(I):
NEXT I
14100 A$=" ":P=0:PRINT "█";:LL=LL+X:PL=P
L+X
14110 SS=PL-7:IF LL-PL<8 THEN SS=LL-15
14120 PRINT "██████████████████████";A$;
"█";:IF SS<0 THEN SS=0
14130 FOR I=SS TO SS+15:PRINT "█";TEXT$(
I);:IF LEN(TEXT$(I))<40 THEN PRINT
14140 IF I=PL-1 THEN PRINT CHR$(62)
14150 NEXT I:PRINT "██
                                   ":RETURN
```

Lo scopo di questo modulo è quello d'inserire la linea corrente nel corpo principale del testo.

Commenti

Linea 14040: se la linea da inserire è più corta di 40 caratteri, viene posta nella posizione indicata dal cursore > di edit.

Linee 14050-14070: se la linea da inserire è più lunga di 40 caratteri, queste istruzioni compiono una ricerca all'indietro dell'ultima parola che entrerà completamente in una linea e trasformeranno tutto ciò che si trova alla sinistra di questa parola nella prima linea da inserire e memorizzare in TT$. A$ viene ora posta uguale alla parte rimanente e il procedimento viene ripetuto. La variabile X registra il numero di linee risultanti.

Linee 14080-14090: il corpo principale del testo, dal cursore di edit in poi, viene spostato per fare posto alle X nuove linee e si provvede al loro inserimento.

Linee 14100-14110: A$ viene posta uguale a un singolo spazio, si azzera la posizione del cursore lampeggiante e il cursore di edit viene posto al disotto delle nuove linee. Si ridefinisce l'inizio della visualizzazione del testo principale in modo da porre il cursore di edit approssimativamente a metà della parte visualizzata.

Collaudo del modulo 1.3.3

Premendo RETURN, dovreste poter inserire linee di testo nel corpo del testo principale.

MODULO 1.3.4

```
15000 REM##############################
15010 REM MOVIMENTO LINEA EDIT
15020 REM##############################
15030 P2=PL-SS
15040 GET T1$:IF T1$<>"" THEN 15070
15050 POKE 54272+FNB(P2),8:POKE FNB(P2),
62:FOR I=1 TO 20:NEXT
15060 POKE FNB(P2),32:GOTO 15040
15070 PL=PL+(T1$="]")+10*(T1$="A"):IF PL
<1 THEN PL=1
15080 PL=PL-(T1$="M")-10*(T1$="B"):IF PL
>LL THEN PL=LL
15090 IF T1$=CHR$(13) THEN RETURN
15100 IF PL>=LL OR T1$<>CHR$(20) THEN 15
120
15110 LL=LL-1:FOR I=PL TO LL:TEXT$(I)=TE
XT$(I+1):NEXT:TEXT$(LL+1)=""
15120 IF PL<LL AND T1$="C" THEN A$=TEXT$
(PL)+" ":RETURN
15130 IF T1$="P" THEN GOSUB 17000
15140 IF T1$="S" THEN GOSUB 18000
15150 IF T1$="F" THEN GOSUB 16000
15160 GOSUB 14110:GOTO 15030
```

Questo modulo permette di spostare il puntatore di edit lungo il testo principale, permettendo così l'inserimento di linee in varie posizioni. Con questo modulo, l'utente viene inoltre messo in grado di chiamare altri moduli per formattare il testo, inviarlo a una stampante o archiviarlo su nastro.

Commenti

Linee 15040-15060: routine a cursore lampeggiante per il cursore principale di edit.

Linee 15070-15080: muovono il cursore di edit verso l'alto o verso il basso. Spostamenti di una sola linea si ottengono tramite le solite frecce per il movimento del cursore. Premendo A o B avrete invece un salto di 10 linee. Da notare l'uso delle condizioni logiche per l'esecuzione di questi movimenti. L'espressione (T1$ = ''A'') vale zero se la condizione non è verificata e meno uno se è vera, si può dunque usare per sostituire in modo pratico un'istruzione IF del tipo IF T1$ = ''A'' THEN ecc.

Linea 15090: premendo RETURN si ritorna al modulo di introduzione del testo.

Linee 15100-15110: premendo DELETE si elimina la linea sotto il cursore.

Linea 15120: premendo C si copia la linea sotto il cursore sulla parte inferiore dello schermo per ulteriori correzioni.

Collaudo del modulo 1.3.4

A questo punto dovreste poter muovere il cursore di edit principale, cancellare delle linee e ricopiarle sul lato inferiore dello schermo.

MODULO 1.3.5

```
16000 REM*******************************
16010 REM FORMATTAZIONE
16020 REM*******************************
16030 FOR I=1 TO LL-2: IF TEXT$(I)="" OR
TEXT$(I+1)="" THEN 16120
16040 SP=39-LEN(TEXT$(I)):FOR J=1 TO LEN
(TEXT$(I+1))
16050 IF MID$(TEXT$(I+1),J,1)<>" " THEN
NEXT J:J=J-1
16060 IF SP<J OR J=LEN(TEXT$(I+1)) THEN
16090
16070 TEXT$(I)=TEXT$(I)+" "+LEFT$(TEXT$(
I+1),J-1)
16080 TEXT$(I+1)=MID$(TEXT$(I+1),J+1):GO
TO 16040
16090 IF LEN(TEXT$(I+1))>=SP THEN 16120
16100 TEXT$(I)=TEXT$(I)+" "+TEXT$(I+1)
16110 FOR J=I+1 TO LL:TEXT$(J)=TEXT$(J+1
):NEXT J:LL=LL-1:PL=PL-1:GOTO 16040
16120 NEXT I:RETURN
```

Questo modulo provvede a formattare il testo il quale, cioè, verrà rimaneggiato in modo che, qualora la cosa sia possibile, gli spazi terminali di una linea vengano riempiti con parole estratte dalla linea seguente.

Linea 16030: quando nel corpo del testo viene inserita una linea vuota, questa non verrà formattata. Si possono così utilizzare delle linee vuote in funzione di separazione di paragrafi o di altre linee che l'utente non desideri avere collegate.

Linee 16040-16080: si calcola lo spazio libero alla fine della linea e si valuta se all'inizio della successiva sia presente una parola che possa entrare in questo spazio — in caso positivo questa viene trasferita.

Linee 16090-16110: se l'intera linea successiva può essere contenuta al termine della linea corrente, verrà aggiunta di seguito e TEXT$ compattata per riempire lo spazio creatosi.

Collaudo del modulo 1.3.5

Scrivendo nel corpo del testo una serie di linee costituite da una sola parola, dovreste ora poter accedere al modo edit del testo, premere F e vedere quindi le parole raccogliersi in linee senza soluzione di continuità. Potete anche inserire brevi linee nel mezzo del testo per poi riformattarlo.

MODULO 1.3.6

```
17000 REM#*******************************
17010 REM USCITA SU STAMPANTE
17020 REM********************************
17030 OPEN 1,4:X=1
17040 IF X=LL THEN 17100
17050 IF TEXT$(X)="" THEN PRINT#1,"":X=X
+1:GOTO 17040
17060 PRINT#1,TEXT$(X);" ";
17070 IF X+1=LL THEN 17100
17080 PRINT#1,TEXT$(X+1):IF TEXT$(X+1)="
" THEN PRINT#1,""
17090 X=X+2:GOTO 17040
17100 PRINT#1,"":CLOSE1:RETURN
```

Questo semplice modulo apre un canale di comunicazione con la stampante (dispositivo 4) e stampa il testo principale. Il testo viene stampato in un formato di 80 colonne (cioè due linee dello schermo costituiscono una linea di stampa), viene inoltre stampata una linea vuota dovunque nel testo originale sia presente una linea vuota. Si noti che, benché il programma stesso sia perfettamente in grado di trattare caratteri minuscoli (premete contemporaneamente i tasti SHIFT e COMMODORE), la maggior parte delle stampanti necessita di un apposito comando per stampare effettivamente caratteri minuscoli. Questo non è stato aggiunto, dal momento che differisce caso per caso. Il manuale della stampante in vostro possesso vi darà tutte le informazioni necessarie.

MODULO 1.3.7

```
18000 REM*********************************
18010 REM FILE DATI
18020 REM*********************************
18030 PRINT "POSIZIONARE IL NASTRO, Q
UINDI PREMERE RETURN--"
18040 INPUT "IL MOTORE SI ARRESTA IN MOD
O AUTOMATICO:";Q$:POKE 192,7:POKE 1,39
18050 PRINT "POSSIBILI FUNZIONI:":PRINT
"1)SALVATAGGIO DATI"
18055 PRINT "2)CARICAMENTO DATI"
18060 INPUT "PREGO SCEGLIERE:";Q:ON Q
GOTO 18080,18120
18070 RETURN
18080 POKE 1,7:FOR I=1 TO 2000:NEXT
18090 OPEN 1,1,2,"TESTI":PRINT#1,PL:PRIN
T#1,LL
18100 FOR I=0 TO LL:PRINT#1,TEXT$(I):NEX
T I
18110 CLOSE 1:RETURN
18120 OPEN 1,1,0,"TESTI":INPUT#1,PL,LL
18130 FOR I=0 TO LL:TEXT$(I)=""
18140 GET#1,A$:IF A$<>CHR$(13) THEN TEXT
$(I)=TEXT$(I)+A$:GOTO 18140
18170 NEXT I:CLOSE1:RETURN
```

Un normale modulo di file dei dati.

Riepilogo

Le tecniche di cui è stato fatto uso in questo programma, per la manipolazione di qualcosa che state osservando sullo schermo, giustificano un certo studio, dal momento che rappresentano il modo più facile (per l'utente) di modificare delle stringhe. Si possono quindi utilizzare in tutti quei programmi in cui, una volta introdotti dei dati del tipo stringa, si vogliano apportare dei cambiamenti — compresi, se lo desiderate, la maggior parte dei programmi di questo libro.

Testi: lista di tasti-istruzione

Modo introduzione testo:
I caratteri del testo si introducono nella posizione del cursore lampeggiante.
Le frecce a destra e a sinistra muovono il cursore lungo la stringa.
← porta il cursore all'inizio o alla fine di una linea.
↑ chiama il modulo di edit.
RETURN pone la stringa corrente all'interno del testo principale.

Modo edit:

RETURN	Ritorna al modo precedente
A, B	Muovono il cursore di edit di 10 linee in alto o in basso
DELETE	Elimina la linea sotto il cursore di edit
C	Copia la linea sotto il cursore di edit
P	Invia il testo alla stampante
S	Salva il testo sul nastro
F	Esegue la formattazione del testo

2. Ausili alla programmazione

Dopo aver visto alcune caratteristiche del 64, ci scostiamo ora momentaneamente dalla struttura normale di questo libro per presentare tre programmi, estremamente compatti, che vi forniranno degli strumenti essenziali per collegare o fondere programmi diversi, rinumerarli e cancellarne agevolmente intere sezioni. I programmi sono molto compatti per il semplice fatto che sono pensati per essere tra loro collegati, tramite la routine MERGE stessa, e aggiunti in coda a programmi esistenti senza occupare troppo spazio in memoria. Una volta terminato di aggiungere ulteriori sezioni ai vostri programmi e di rinumerarli, la routine DELETE non avrà difficoltà a cancellare se stessa e i suoi due compagni!

2.1 Merge

Questo programma, insieme agli altri due presentati in questo breve capitolo, è assolutamente indispensabile per coloro i quali intendano considerare in modo serio la programmazione modulare. Tramite questo piccolo programma, si possono risparmiare ore di lavoro, conservando su nastro dei moduli di una certa utilità e semplicemente collegandoli tramite l'azionamento di un pulsante del registratore. Nel presentare il programma, sono in debito con Steve Beats della Commodore inglese, il quale mi ha suggerito l'idea fondamentale da cui è stato sviluppato.
Nel programma non si hanno moduli — con poche linee, difficilmente ne sarebbe valsa la pena — e tuttavia un simile programma metterà a vostra disposizione la programmazione modulare.
Quello che fa, è prendere un altro programma (o gruppi di programmi o singole sezioni) dal nastro e inserirlo nel 64 senza pericolo di perdere ciò che già vi si trovi — a meno che non coincidano i numeri di linea, nel qual caso il primo programma verrà modificato.

MODULO 2.1.1

```
61000 END
63980 INPUT"NOME PROGRAMMA";N$:INPUT" #
MODULI";N:POKE679,N
63981 LL=LEN(N$):POKE680,LL:FORX=1TOLL:P
OKE680+X,ASC(MID$(N$,X,1)):NEXT
```

```
63982 POKE2,128:T=PEEK(679):IF T=0 THEN
POKE198,0:END
63983 POKE679,T-1:LL=PEEK(680):N$="":FOR
X=1TOLL:N$=N$+CHR$(PEEK(680+X)):NEXT
63984 N1$=N$+RIGHT$(STR$(T),1):OPEN1,1,0
,N1$:A$=""
63986 GET#8,A$:IF A$=CHR$(13)THEN63986
63987 PRINT"□■":PRINTA$;:GOTO63991
63990 POKE184,1:POKE185,96:POKE186,1:POK
E152,1:PRINT"□■":PRINTCHR$(PEEK(2));
63991 GET#8,A$
63992 PRINTA$;:IF A$<>CHR$(13)THEN63991
63993 GET#8,A$:POKE2,ASC(A$)
63994 IFA$=CHR$(13)ORA$=CHR$(10)THENCLOS
E1:PRINT"GOTO63982▒";:GOTO 63997
63995 PRINT"GOTO63990▒";
63997 POKE631,13:POKE632,13:POKE633,13:P
OKE198,3
```

Qualora si operi il caricamento di un programma dal nastro nel modo consueto, un eventuale altro programma già residente in memoria verrà cancellato: non è dunque possibile utilizzare LOAD per collegare più programmi.

Iniziamo tuttavia con l'osservare che, per introdurre una linea di programma nel 64, è sufficiente scriverla sullo schermo, posizionare il cursore in un punto qualsiasi di questa linea, quindi premere RETURN. Questa semplicità ha come contropartita il fatto che la macchina automaticamente chiuderà qualsiasi file aperto, perderà sia i nomi che i valori delle variabili di qualsiasi tipo (stringhe, interi, matrici, ecc.), perderà gli indirizzi di ritorno delle subroutine, dimenticherà cicli già iniziati, ecc.

Memorizzando su nastro i programmi da salvare tramite i comandi OPEN 1,1,2,"NOME": CMD1: LIST e chiudendo correttamente il file tramite PRINT#1: CLOSE1, in virtù del comando CMD1, invieremo al nastro il listato di un programma correntemente in memoria sotto la forma della sequenza di caratteri ASCII che si vedrebbe sullo schermo se venisse listato. Il file di caratteri così creato si chiamerà "NOME". (Nella pratica converrà usare nomi tipo MODULO8, MODULO7, ecc., formati cioè da un nome seguito da un numero, curando di utilizzare numeri decrescenti, dal momento che ciò non creerà problemi di ricerca sul nastro — con il disco, il problema non si pone).

Le considerazioni precedenti suggeriscono la struttura della routine di fusione, che aprirà un file di lettura verso il nastro, leggerà i caratteri relativi a ogni linea e li stamperà sullo schermo. Raggiunto il termine della linea, vi posizionerà il cursore e "premerà" RETURN. Occorrerà altresì fare in modo che la routine non dimentichi le variabili necessarie al suo funzionamento e mantenga aperti i file.

Commenti

Linea 61000: dal momento che questo programma, come i rimanenti due di questo capitolo, è pensato per risiedere "in coda" a qualsiasi altro programma si trovi in macchina, è bene proteggerlo da programmi precedenti non correttamente terminati. Per lanciare una procedura di fusione, battere GOTO 63980.

Linea 63980: si domanda all'utente di specificare il nome, comune a tutti i programmi da leggere dal nastro, e il numero di questi. Verranno caricati in sequenza tutti i programmi da NOME (numero più elevato) a NOME1 (ad esempio da MODULO8 a MODULO1, per numeri decrescenti).

Linea 63981: vengono salvati in un'area libera, all'inizio della RAM, il numero dei moduli da fondere (all'indirizzo 679), la lunghezza del nome dei moduli (indirizzo 680), e il nome stesso, carattere per carattere (da 681 a 681 + LL). L'uso delle POKE si giustifica per la necessità di proteggere queste informazioni in modo diverso da una normale variabile, che andrebbe persa all'introduzione di ogni nuova linea di programma.

Linea 63982: l'indirizzo 2, normalmente inutilizzato, servirà in seguito: gli si attribuisce un valore iniziale. Tutte queste aree di memoria non sono interessate né dal programma, né da variabili, variabili di sistema, buffer o altro: rimarranno quindi indisturbate qualsiasi cosa accada. Di seguito, il criterio di terminazione: il contenuto dell'indirizzo 679 viene decrementato di 1 (alla linea seguente) ogniqualvolta si inizia il caricamento di un nuovo modulo; raggiunto lo 0 il programma termina.

Linee 63983-63984: si decrementa il contenuto dell'indirizzo 679 (il numero dei moduli ancora da caricare) e si ricostruisce il nome del modulo; gli si appone il numero d'ordine raggiunto e si apre verso il registratore a cassette un file di quel nome (N1$).

Linea 63986: all'inizio del file si trovano un certo numero di CHR$(13), che non ci interessano.

Linee 63991-63992: si raccolgono dal nastro e si stampano sullo schermo i caratteri del listato precedentemente memorizzato. Un CHR$(13) indica la fine della linea e il luogo in cui era stato premuto RETURN.

Linea 63993: se la routine è giunta a questo punto, ha rivelato la presenza di un carattere di RETURN che, semplicemente stampato, non fa altro che mandare a capo la posizione di stampa. Questa linea raccoglie comunque il carattere successivo presente sul nastro e lo salva nella locazione 2 della RAM.

Linea 63995: normalmente la condizione alla linea 63994 è falsa, dunque la routine giunge a questa linea. Essendo andata a capo per la stampa del CHR$(13) raccolto dal nastro, la routine provvede a stam-

pare sullo schermo GOTO 63990 e il carattere di riposizionamento del cursore. Lo schermo ha ora questo aspetto: in alto a sinistra, il cursore; alla linea seguente, la linea di programma che vorremmo caricare; alla linea immediatamente seguente, la scritta GOTO 63990. Se ora la routine terminasse, premendo RETURN una prima volta, porteremmo il cursore sulla linea di programma da introdurre; una seconda volta, introdurremmo la linea stessa e il cursore verrebbe posto sulla scritta GOTO 63990. A questo punto avremmo perso le variabili e chiusi i file. Un terzo azionamento di RETURN eseguirebbe il comando GOTO indicato.

Linea 63997: vengono forzati nella coda del buffer della tastiera i codici "13" relativi al tasto di RETURN e, all'indirizzo 198, il numero di tasti non ancora valutati da parte del sistema. L'effetto di queste POKE è dunque equivalente a tre azionamenti di RETURN. Solamente dopo la fine del programma il sistema va ad esaminare il buffer della tastiera. Dopo questa linea, tuttavia, la routine termina e il sistema "preme" RETURN per tre volte, introducendo così la linea di programma visualizzata e causando l'inizio di un nuovo ciclo a 63990.

Linea 63990: queste POKE servono a superare l'inconveniente della chiusura dei file, indicando al sistema che il file logico 1, di lettura, verso il nastro, è ancora aperto (cfr. la mappa della memoria del 64). Viene inoltre stampato il primo carattere seguente il CHR$(13), che era stato salvato nella locazione 2.

Linea 63994: se il programma da fondere è terminato, dopo il CHR$(13) di fine linea si trova un CHR$(10) marcatore di end-of-file. Sarà dunque necessario chiudere correttamente il file[1], introdurre l'ultima linea visualizzata (come in precedenza) e attivare la ricerca del nuovo modulo da caricare, reiniziando cioè alla linea 63982.

---

(1) Nel caso del disco, il programma si modifica, per esempio, come segue:

```
61000 END
63980 INPUT"NOME PROGRAMMA";N$:INPUT" #
MODULI";N:POKE679,N
63981 LL=LEN(N$):POKE680,LL:FORX=1TOLL:P
OKE680+X,ASC(MID$(N$,X,1)):NEXT
63982 POKE2,128:T=PEEK(679):IF T=0 THEN
POKE198,0:END
63983 POKE679,T-1:LL=PEEK(680):N$="":FOR
X=1TOLL:N$=N$+CHR$(PEEK(680+X)):NEXT
63984 N1$=N$+RIGHT$(STR$(T),1):OPEN8,8,2
,N1$:A$=""
63986 GET#8,A$:IF A$=CHR$(13)THEN63986
63987 PRINT"▢▣":PRINTA$;:GOTO63991
63990 POKE184,8:POKE185,2:POKE186,8:POKE
152,1:PRINT"▢▣":PRINTCHR$(PEEK(2));
63991 GET#8,A$
63992 PRINTA$;:IF A$<>CHR$(13)THEN63991
63993 GET#8,A$:POKE2,ASC(A$)
63994 IFA$=CHR$(13)ORA$=CHR$(10)THENPRIN
T#8:CLOSE8:PRINT"GOTO63982▣";:GOTO 63997
63995 PRINT"GOTO63990▣";
63997 POKE631,13:POKE632,13:POKE633,13:P
OKE198,3
```

e la chiusura del file di lettura risulta indispensabile.
Sempre nel caso del disco, i programmi andranno salvati con OPEN8,8,2, "0: NOME, SEQ, W": CMD8: LIST. Il file andrà poi chiuso con PRINT # 8: CLOSE8.

È necessario l'uso di un nastro di buona qualità, dal momento che dei difetti di riproduzione bloccheranno il programma. È necessario altresì che la versione listata delle linee di programma da fondere non superi la lunghezza di due linee di schermo, altrimenti si avranno messaggi di SYNTAX ERROR: evitate dunque un uso esasperato delle abbreviazioni.

2.2 Delete

Quando si sviluppano programmi che impiegano moduli simili a quelli di programmi registrati in precedenza, una cosa utile è poter caricare il programma originale, per cancellare solo quelle parti non più necessarie alla nuova applicazione. Questa routine di 12 linee vi metterà in grado di fare esattamente questo.
La routine si basa sul modo estremamente chiaro e semplice in cui i listati del programma sono disposti nella memoria dei computer Commodore. Ogni linea di programma nella memoria inizia con due byte di collegamento (*link*) che specificano l'indirizzo in memoria di inizio della linea successiva. Questi sono seguiti da due byte che registrano il numero di linea effettivo. La nostra routine scandisce i numeri di linea tra un valore iniziale e uno finale, entrambi specificati dall'utente. Quando si trova l'indirizzo dell'ultima linea da eliminare, il programma pone semplicemente l'indirizzo della linea successiva nella prima delle linee da cancellare, per puntare alla linea immediatamente seguente l'ultima linea da eliminare. L'effetto complessivo è di produrre una singola linea che si estende dall'inizio della prima linea da cancellare alla fine dell'ultima.
Può ora avere luogo la cancellazione, semplicemente cancellando la prima linea — tutte le altre la seguono.

MODULO 2.2.1

```
61000 END
63700 INPUT"PRIMA LINEA DA CANCELLARE:";
C1
63710 INPUT"ULTIMA LINEA DA CANCELLARE:"
;C2
63715 DEF FNDH(X)=PEEK(X)+256*PEEK(X+1)
63716 DEF FNH1(X)=X AND 255
63717 DEF FNH2(X)=INT(X/256)
63720 LA=2049
63730 LN=FNDH(LA+2):IF LN<C1 THEN LA=FND
H(LA):GOTO 63730
63740 DSTART=LA
63750 LN=FNDH(LA+2):IF FNDH(LA)=0 THEN 6
3760
63755 IF LN<=C2 THEN LA=FNDH(LA):GOTO 63
750
63760 POKE DSTART,FNH1(LA):POKE DSTART+1
,FNH2(LA)
63770 POKE DSTART+4,143:FOR I=5 TO 10:PO
KE DSTART+I,33:NEXT
63780 END
```

Commenti

Linea 63715: questa funzione, che può risultare utile in svariati contesti, converte un numero di due byte, del tipo di quelli trattati dalla più parte dei calcolatori, in un comune numero decimale compreso tra 0 e 65535. Il numero di due byte ha in realtà base 256, cioè è composto da una cifra dotata di un peso massimo di 255 unità e da una seconda cifra che vale fino a 255 * 256, nello stesso modo per cui 99 vale 9 unità più 9 volte 10. Tuttavia, tanto per confondere le idee, le cifre vengono memorizzate a rovescio, con il byte di peso più elevato in seconda posizione.

Linee 63716-63717: queste due funzioni-utente compiono l'operazione inversa, di trasformare un numero decimale in due byte.

Linea 63720: LA viene posto uguale all'indirizzo iniziale della prima linea del programma.

Linea 63730: LN viene posto uguale al valore del terzo e quarto byte della linea — il numero di linea — e se questo è inferiore al valore della prima linea da cancellare, allora LA utilizza i due byte di collegamento per saltare all'indirizzo iniziale della linea successiva. Il procedimento si ripete fino a trovare un numero di linea maggiore o uguale a quello della prima linea da cancellare.

Linea 63740: l'indirizzo iniziale della prima linea da cancellare viene memorizzato nella variabile DSTART.

Linea 63750: tramite la funzione FNDH, la variabile LA scorre la memoria da un inizio di linea al successivo e in ogni salto la variabile LN viene eguagliata al numero di linea incontrato. Se FNDH trova una coppia di byte contenenti zero, dove dovrebbe trovarsi la coppia dei byte di collegamento, si è raggiunta la fine del programma.

Linea 63755: ogni volta che si incontra un nuovo numero di linea, viene confrontato con quello dell'ultima linea da cancellare. Se non è stata raggiunta l'ultima linea, si compie il salto successivo.

Linea 63760: se il programma ha raggiunto questo punto, ha trovato l'ultima linea da cancellare e pone con la POKE l'indirizzo della linea successiva nei due byte di collegamento di DSTART.

Linea 63770: il primo carattere del nuovo blocco di una sola linea da cancellare viene trasformato in una REM e di seguito a questa vengono forzati una serie di punti esclamativi per segnare la linea da cancellare.

La routine ha ora completato il suo lavoro, basta inserire, da tastiera, il numero della linea segnata dalla REM e premere RETURN — tutto il blocco, da poche linee a un intero programma, sparirà.
In pratica, questa routine trova il suo migliore utilizzo in unione alla routine di fusione del paragrafo precedente (caricate la routine MERGE e aggiungete questa, oppure collegatele insieme per mezzo

della MERGE stessa), dato che ciò permetterà di aggiungere la routine a programmi esistenti dai quali volete estrarre alcune linee evitandone altre. Impiega pochi istanti di esecuzione e può far risparmiare un sacco di tasti da premere.

2.3 Renumber

Una cosa che tutti desiderano, è avere dei programmi numerati in modo ordinato — in qualche modo questo fa la differenza tra un programma di aspetto professionale e un altro che appare completamente trascurato. Usando il programma relativamente breve presentato in questo paragrafo, potete rinumerare a piacimento, benché non si possa proprio dire che sia molto rapido e imponga qualche limitazione al campo dei numeri di linea ammessi.
La routine rinumererà qualsiasi programma, comprese le istruzioni GOTO, GOSUB, ON... GOTO e ON... GOSUB e i numeri di linea seguenti IF... THEN. Quello che non fa è rilocare il programma in memoria: perciò non può aggiungere o togliere cifre a GOTO (ecc.), dal momento che questo significa spostare tutta la parte di programma seguente gli indirizzi modificati. Non che ciò sia impossibile, o particolarmente difficoltoso, solo che per avere risultati soddisfacenti bisognerebbe far ricorso al linguaggio macchina, cosa che esula dai limiti della presente trattazione.
Dipende da questa limitazione il fatto che tutti i programmi di questo libro, che sono stati rinumerati con questa routine (dove i numeri di linea sono irregolari, è a causa di modifiche introdotte successivamente), inizino a 11000, assicurando in tale modo che tutti i numeri di linea abbiano cinque cifre.
Il modo in cui è strutturato il programma rinumerato si può controllare usando linee per modificare il formato all'interno del programma sorgente. Un'attenta osservazione dei programmi di questo libro mostrerà che i moduli iniziano praticamente sempre con una REM e che il primo carattere dopo la REM è il simbolo "#".
Il programma di rinumerazione è concepito in modo da iniziare a 11000 e continuare in passi di 10 fino al raggiungimento di una linea del tipo anzidetto, dove si incrementa il numero di linea al migliaio successivo.
Non solo ciò rende i programmi di agevole lettura, ma significa anche che potete controllare la struttura del programma da rinumerare. Per esempio, disponete di un programma del quale volete utilizzare tre o quattro moduli, ma gli attuali numeri di linea non sono confacenti alla struttura che desiderate per il nuovo programma — a causa, supponiamo, di insufficiente spazio tra i moduli per fonderli con qualcos'altro che avete sul nastro. Inserendo due istruzioni REM il cui primo carattere sia "#" ed eseguendo la rinumerazione, aprite automaticamente uno spazio di 2000 dove si trovano le REM — per niente complicato!

MODULO 2.3.1

```
63000 CLR:DIM ZZ(500,1):LA=2049:PP=LA
63010 DEF FNDH(X)=PEEK(X)+256*PEEK(X+1)
63013 DEF FNH1(X)=X AND 255
63016 DEF FNH2(X)=INT(X/256)
```

```
63050 IF PP<>FNDH(LA) THEN 63060
63053 LA=FNDH(LA)::NL=FNDH(LA+2):IF NL=6
3000 THEN GOTO 63500
63058 IF PEEK(LA+5)=143 THEN PP=FNDH(LA)
:GOTO 63053
63059 PP=PP+4
63060 IF PEEK(PP)<>167 THEN 63070
63062 S=0:IF PEEK(PP+1)=32 THEN S=1
63064 IF PEEK(PP+1+S)<48 OR PEEK(PP+1+S)
>57 THEN 63200
63065 GOTO 63076
63070 IF PEEK(PP)<>137 AND PEEK(PP)<>141
 THEN 63200
63076 LET S=0:IF PEEK(PP+1)=32 THEN S=1
63080 GG$="":FOR I=1+S TO 5+S:IF PEEK(PP
+I)<48 OR PEEK(PP+I)>57 THEN GOTO 63140
63085 LET GG$=GG$+CHR$(PEEK(PP+I)):NEXT
63090 GG=VAL(GG$):L1=2049:L2=FNDH(2051):
LL=11000
63093 IF L2=63000 THEN PRINT"NUMERO DI L
INEA NON DEFINITO A";NL:STOP
63095 IF L2=GG THEN 63100
63097 L1=FNDH(L1):L2=FNDH(L1+2):LL=LL+10
63098 IF PEEK(L1+4)=143 AND PEEK(L1+5)=3
5 THEN LL=1000*INT((LL+1000)/1000)
63099 GOTO 63093
63100 LET ZZ(ZI,0)=LL:LET ZZ(ZI,1)=PP+S:
ZI=ZI+1
63110 IF PEEK(PP+S+6)=44 THEN PP=PP+S+6:
GOTO 63076
63135 GOTO 63200
63140 PRINT"COMANDO NON STANDARD ALLA LI
NEA ";NL:STOP
63200 PP=PP+1:GOTO 63050
63500 LA=2049:LL=10000
63503 IF PEEK(LA+4)=143 AND PEEK(LA+5)=3
5 THEN LL=1000*INT((LL+1000)/1000)
63505 IF FNDH(LA+2)=63000 THEN 63600
63510 POKE LA+2,FNH1(LL):POKE LA+3,FNH2(
LL)
63520 LET LL=LL+10:LET LA=FNDH(LA):GOTO
63503
63600 IF ZI=0 THEN STOP
63605 FOR I=0 TO ZI-1
63610 FOR J=1 TO 5:POKE ZZ(I,1)+J,ASC(MI
D$(STR$(ZZ(I,0)),J+1)):NEXT J
63620 NEXT I
63650 STOP
```

Commenti

Linea 63000: la matrice ZZ verrà utilizzata per registrare gli indirizzi delle GOTO, ecc., che devono essere rinumerate e i nuovi numeri da assegnare loro.

Linee 63010-63016: stesse funzioni come nella routine DELETE. Dovendo fondere le due, sarà necessario eliminare questa parte.

Linee 63050-63059: PP è un puntatore che scandisce la memoria alla ricerca di GOTO ecc. Comincia all'inizio dell'area di programma a 2049. Ogni volta che raggiunge l'inizio di una nuova linea, viene incrementata la variabile LA di indirizzo di linea. Il numero di linea della linea corrente viene memorizzato in LN e il programma finisce di lavorare alla linea 63000 — l'inizio di questa routine. A questo punto PP salta al primo carattere della linea.

Linee 63060-63065: ogniqualvolta viene incontrato un THEN in memoria, queste linee controllano se questo sia seguito, o subito dopo o intervallato da uno spazio, da un numero di linea. La variabile S registra semplicemente la presenza dello spazio. Se il THEN non è seguito da un numero, PP continua.

Linee 63070-63085: se viene trovato il codice di una GOSUB o di una GOTO, il programma esegue un controllo per stabilire la presenza o meno di uno spazio. Si costruisce CG$ con le cifre delle linee di destinazione. Meno di cinque cifre producono un messaggio d'errore alla linea 63140.

Linee 63090-63099: GG viene posto uguale alla destinazione della GOTO o GOSUB. La routine ora scandisce i numeri di linea del programma, partendo dall'inizio, in cerca della destinazione. Per ogni linea esaminata, la variabile LL viene incrementata di 10, cominciando da 11000, e registra così quale sarà il numero di linea una volta rinumerato il programma — la linea non può essere rinumerata a questo punto, dal momento che un'altra GOTO potrebbe riferirvisi. Quando si incontra una REM#, LL si incrementa fino al migliaio successivo.

Linea 63100: a questo punto è stato trovato il numero di linea esatto, così l'indirizzo della GOTO viene memorizzato nella matrice ZZ, insieme al numero di linea che assumerà (LL).

Linea 63110: se il sesto carattere dopo la GOTO o GOSUB è una virgola, si suppone che si tratti di una ON... GOTO/GOSUB, PP viene fatto avanzare e la nuova linea di destinazione viene raccolta da un precedente spezzone della routine.

Linea 63200: il procedimento continua, con PP che si sposta lungo la memoria fino a incontrare la linea 63000.

Linee 63500-63520: queste linee cominciano all'inizio del programma e ne rinumerano solo le linee (tenendo conto delle REM#).

Linee 63600-63650: tutto ciò che resta da fare è estrarre da ZZ l'indirizzo di tutte le destinazioni delle GOTO e GOSUB e inserire le nuove destinazioni nei cinque byte che seguono ogni indirizzo — queste sono già state calcolate.

Benché questo programma non regga il confronto per velocità o flessibilità con una buona routine in linguaggio macchina, svolge il suo compito, come testimoniano i programmi di questo libro. Se non possedete una routine di rinumerazione in codice macchina, prevedo che ritornerete a questa più spesso di quanto vi succederà con praticamente ogni altro programma.
Una volta fusa con le due precedenti routine, isolate l'una dall'altra da istruzioni di STOP, vi sarete costituiti un potente strumento, multifunzionale, in grado di rendere più piacevole la programmazione e più presentabili i vostri programmi.

3. Il Commodore 64 a colori

Il Commodore 64 mette a disposizione un insieme pressoché stupefacente di possibilità grafiche. Le forme e i colori che può disegnare sono sufficienti per far fronte praticamente a qualsiasi necessità, e certamente bastano per tenere occupato l'artista dilettante per tutta la vita. In questo capitolo troverete quattro programmi che vi metteranno in grado di esplorare il mondo dei caratteri grafici, dei caratteri-utente, delle animazioni e della grafica in bit-map. Quattro soli programmi non possono assolutamente mettere la parola fine a quello che si può ottenere dal 64, perciò i programmi sono progettati come strumenti, il cui scopo è di permettervi di inserire nei vostri successivi programmi tutte quelle multicolori caratteristiche che li solleveranno dalla banalità.

3.1 Artista

Pochi micro domestici hanno un insieme di caratteri grafici pari a quello delle macchine Commodore. È difficile pensare a qualcosa che non si possa disegnare in qualche forma o maniera usando le combinazioni di caratteri disponibili da tastiera. Ciò risulta estremamente utile per vivacizzare i programmi, specialmente in combinazione con le eccellenti capacità di gestione del colore del 64.
Una limitazione a tutto ciò si trova nel processo creativo di sviluppo pratico di figure grafiche. Naturalmente questo si può ottenere con istruzioni PRINT nel programma, ma ottenere proprio le istruzioni desiderate, con una quantità di comandi di colore, comandi di inversione e così via, dovendo definire separatamente ogni linea, può diventare un lavoro estremamente noioso. Ciò che serve è un modo di trattare lo schermo a mo' di cavalletto, dipingendovi caratteri grafici in svariati colori, cancellando, modificando a piacere, per poi, ad uso della posterità (o per lo meno ad uso di altri programmi che potrebbero utilizzare il disegno creato), salvare il disegno su nastro.
Il programma seguente cerca di fare tutto ciò.
Lungo la strada raccoglierete una discreta quantità d'informazioni su come manipolare la memoria di colore e di schermo, insieme a utili locazioni di memoria per controllare caratteristiche come il colore di stampa.

Artista: lista delle variabili

CC	Posizione corrente del cursore
CO(3)	Coordinate di due angoli del disegno da salvare
CT	Memorizzazione temporanea della posizione del cursore
CU	Colore corrente del cursore
D1$	Valori dei caratteri del disegno da salvare
D2$	Valori dei colori dei caratteri da salvare
D3$, D4$	Copie temporanee di D1$ e D2$
MODO	Stabilisce quale caratteristica del colore viene indirizzata
PC	Colore del carattere alla posizione PP
PP	Contenuti originali della locazione corrente del cursore
PT	Locazione in memoria dei due angoli contenuti nella matrice CO

MODULO 3.1.1

```
11000 REM#*******************************
11010 REM VARIABILI
11020 REM********************************
11030 R$=CHR$(13)
```

Piuttosto eccessivo chiamarlo modulo, ma la sua presenza fa sì che, se decidete ulteriori sviluppi del programma, sia disponibile un'area separata per le necessarie variabili. La stringa effettivamente definita è un separatore standard nei file dei dati.

MODULO 3.1.2

```
12000 REM#*******************************
12010 REM CURSORE,MOVIMENTO,STAMPA
12020 REM********************************
12030 PRINT"⊐";
12040 POKE 650,128:GET A$
12050 CC=PEEK(211)+256*PEEK(210)+PEEK(20
9):PP=PEEK(CC):PC=PEEK(CC+54272)
12060 POKE CC,42:POKE CC+54272,CU:FOR I=
1 TO 15:NEXT:POKECC,PP:POKECC+54272,PC
12070 IF A$="" THEN 12040
12080 IF CC>1983 AND A$="▩" THEN 12040
12090 IF CC=2023 AND A$="▐" THEN 12040
12100 IF CC=1024 AND A$="▌" THEN 12040
12110 IF CC<1064 AND A$="⌝" THEN 12040
12120 IF A$="⌝" OR A$="▩" OR A$="▌" OR A
$="▐" THEN PRINT A$;:GOTO 12040
12130 IF A$=CHR$(133) THEN MODO=1
12140 IF A$=CHR$(137) THEN MODO=2
12150 IF A$=CHR$(134) THEN MODO=3
12155 IF A$=CHR$(138) THEN 12030
12160 IF (MODO=1 OR MODO=2 OR MODO=3) AN
D A$="←" THEN MODO=MODO+.5:GOTO 12040
12170 IF A$=CHR$(135) THEN INV=(INV=0)
12180 IF A$=CHR$(139) THEN MODO=6
12190 IF A$=CHR$(136) THEN MODO=7
```

```
12200 IF A$=CHR$(140) THEN MODO=8
12210 IF MODO=1 AND A$>="1" AND A$<="8"
THEN POKE CC+54272,VAL(A$)-1:GOTO 12040
12220 IFMODO=1.5ANDA$>="1"ANDA$<="8" THE
N POKECC+54272,8+VAL(A$)-1:GOTO 12040
12230 IF MODO=2 AND A$>="1" AND A$<="8"
THEN POKE53281,VAL(A$)-1:GOTO 12040
12240 IF MODO=2.5AND A$>="1" AND A$<="8"
 THEN POKE53281,VAL(A$)+8:GOTO 12040
12250 IF MODO=3 AND A$>="1" AND A$<="8"
THEN POKE646,VAL(A$)-1:GOTO 12040
12260 IF MODO=3.5 AND A$>="1" AND A$<="8
" THEN POKE646,8+VAL(A$)-1:GOTO 12040
12270 IF MODO<>6 OR (A$<>"R"AND A$<>"D"A
ND A$<>"S") THEN 12320
12280 IF A$="R" THEN GOSUB 13000
12290 IF A$="D" THEN GOSUB 13060
12300 IF A$="S" THEN GOSUB 14000
12310 CC=CT:GOTO 12040
12320 IF MODO=8 AND A$>="1" AND A$<="8"
THEN CU=VAL(A$)-1:GOTO 12040
12330 IF INV=-1 THEN PRINT"▚";
12340 PRINT A$;"▆";
12350 GOTO 12040
```

Questo modulo è in realtà tutto ciò che serve per trasformare lo schermo in un cavalletto d'artista. Lo scopo è di mettervi in grado di muovere un cursore lampeggiante lungo tutto lo schermo, stampare caratteri, modificarli, cancellarli, scambiare i colori di sfondo e carattere...

Commenti

Linee 12040-12070: questa routine definisce un cursore lampeggiante controllato dall'utente.

Linea 12040: questa POKE introduce la funzione di ripetizione in modo che un tasto, se tenuto premuto, continui a stampare lo stesso carattere. La seconda parte della linea riceve ogni singolo carattere introdotto da tastiera.

Linea 12050: a CC viene assegnato l'indirizzo di memoria della posizione corrente di stampa. PEEK(211) dà la posizione lungo la linea (0-39), mentre PEEK(210) * 256 + PEEK(209) dà l'indirizzo di memoria dell'inizio della linea. PP è posto uguale al codice di schermo di ciò che occupa attualmente la posizione dove sta lampeggiando il cursore. PC è il colore del carattere in quella posizione.

Linea 12060: si inserisce ora un asterisco, codice di schermo 42, nella posizione dove si vuol fare lampeggiare il cursore e il colore corrente del cursore CU viene inserito nella memoria di colore alla posizione corrispondente. Un breve ciclo di temporizzazione mantiene

l'asterisco sullo schermo per un momento, dopodiché vengono riposti nella memoria il carattere (codice PP) e il colore (PC) originali.

Linea 12070: se non è stato premuto alcun tasto, il ciclo viene ripetuto.

Linee 12080-12110: queste linee controllano che il cursore non tenti di uscire dallo schermo quando vengono premute le frecce di movimento cursore.

Linea 12120: se viene inserito un controllo di cursore che superi i test delle 4 linee precedenti, questo viene immediatamente stampato e il programma ritorna alla routine di cursore lampeggiante.

Linee 12130-12200: utilizzando come ingresso i tasti funzione sulla destra della tastiera, queste linee danno all'utente la possibilità di specificare diversi *modi* che permettono d'impostare diverse caratteristiche del colore.

Linea 12130: premendo il tasto f1 si pone il programma nel MODO 1. Di conseguenza, l'azionamento di uno dei tasti da 1 a 8 ridefinirà il colore di ognuno dei caratteri su cui viene posizionato il cursore. Il colore corrisponderà a quello indicato sul davanti del tasto.

Linea 12140: premendo f2 si attiva la stessa procedura per ridefinire il colore di sfondo dello schermo.

Linea 12150: premendo f3 si ottiene la ridefinizione del colore di stampa tramite la stessa procedura.

Linea 12160: in realtà sono disponibili 16 colori. L'azionamento del tasto con la freccia a sinistra (nell'angolo superiore sinistro della tastiera), mentre ci si trova nei modi 1, 2 o 3, ridefinisce il modo in maniera che l'azionamento dei tasti 1-8 fornirà il colore normalmente ottenuto da quel tasto più il tasto Commodore. La funzione ridefinita sarà la stessa del modo principale.

Linea 12170: premendo f5 viene attivata o disattivata la funzione d'inversione — permettendo così di stampare caratteri inversi.

Linea 12180: premendo f6 il programma permetterà di salvare il disegno creato. La procedura dettagliata verrà spiegata in seguito.

Linea 12190: f7 non fa nient'altro che ridefinire un modo non attivo. Ciò permette all'utente di stampare i numeri 1-8, invece di ridefinire una funzione di colore.

Linea 12200: f8 permette all'utente di ridefinire il colore del cursore in uno dei primi otto colori. Particolarmente utile se il colore di schermo è stato trasformato in modo tale da rendere il cursore scarsamente visibile.

Linee 12210-12220: se il MODO è 1 o 1.5, l'ingresso di colore viene posto (POKE) nella locazione di memoria colore del quadratino corrente.

Linee 12230-12240: se il MODO è 2 o 2.5, il nuovo codice di colore viene posto nella locazione 53281, che stabilisce il colore di sfondo dello schermo.

Linee 12250-12260: se il MODO è 3 o 3.5, il nuovo codice di colore viene posto nella locazione 646, che specifica il colore di stampa corrente.

Linee 12270-12310: quando si è nel MODO 6, questa routine consente il salvataggio di disegni grandi e piccoli. Introducendo R si permette la definizione di un rettangolo di schermo da salvare. D salva un disegno di piccole dimensioni, mentre S salva su nastro l'intera videata.

Linea 12310: CT viene utilizzata per salvare la posizione corrente del cursore, che potrebbe essere alterata dalla routine di SAVE.

Linea 12320: se il MODO è 8, il nuovo codice di colore viene memorizzato nella variabile CU.

Linea 12330: se viene impostata la funzione d'inversione (INV = —1) viene stampato il carattere di controllo relativo (RVS ON), invertendo così il prossimo carattere da stampare.

Linea 12340: se il programma è giunto a questo punto, qualunque carattere battuto viene stampato sullo schermo, seguito dal carattere di controllo di esclusione dell'inversione.

Collaudo del modulo 3.1.2

Dopo l'introduzione di questo modulo, dovreste poter disegnare sullo schermo a piacimento, tramite l'intero set di caratteri disponibili da tastiera. Tutti i comandi di ridefinizione dei colori dovrebbero risultare disponibili, ma non sarete ancora in grado di salvare i disegni che realizzate.

MODULO 3.1.3

```
13000 REM#******************************
13010 REM SALVATAGGIO DISEGNI
13020 REM*******************************
13030 GET T$:IF T$="" THEN 13030
13040 IF T$<>"1" AND T$<>"2"THEN RETURN
13050 CO(VAL(T$)*2-2)=PEEK(211):CO(VAL(T
$)*2-1)=INT((CC-1024)/40):RETURN
13060 REM*******************************
13070 CT=CC:POKE 646,CU
13080 IF CO(0)<=CO(2) AND CO(1)<=CO(3) T
HEN GOTO 13110
13090 PRINT "▒RETTANGOLO DEFINITO IN MOD
O IMPROPRIO.";:FOR I=1 TO 1000:NEXT
```

```
13100 PRINT "
               ";:RETURN
13110 IF (CO(2)-CO(0)-1)*(CO(3)-CO(1)-1)
<251 THEN 13140
13120 PRINT"DISEGNO TROPPO GRANDE.";:FO
R I=1 TO 1000:NEXT
13130 PRINT"
               ";:RETURN
13140 FOR I=1 TO 2:PT=1024+40*CO(I*2-1)+
CO(I*2-2):TC(I)=PT:TC(I+2)=PEEK(PT)
13150 POKE PT,42:POKE PT+54272,CU:NEXT
13160 INPUT "PUNTI CORRETTI (S/N):";Q$:
IF Q$="S" THEN 13170
13162 PRINT"
   ":FORI=1TO2:POKE TC(I),TC(I+2):NEXT
13164 RETURN
13170 D1$="":D2$="":FOR I=CO(1)+1 TO CO(
3)-1:FOR J=CO(0)+1 TO CO(2)-1
13180 D1$=D1$+CHR$(PEEK(1024+40*I+J)):PO
KE 1024+40*I+J,42
13190 D2$=D2$+CHR$(PEEK(55296+40*I+J)):P
OKE 55296+40*I+J,0:NEXT J,I:PRINT"";
13200 D3$=D1$:D4$=D2$:FOR I=CO(1)+1 TO C
O(3)-1:FOR J=CO(0)+1 TO CO(2)-1
13210 POKE 1024+40*I+J,ASC(LEFT$(D3$,1))
:D3$=RIGHT$(D3$,LEN(D3$)-1)
13220 POKE 55296+40*I+J,ASC(LEFT$(D4$,1)
):D4$=RIGHT$(D4$,LEN(D4$)-1):NEXT J,I
13230 PRINT "QUESTO E' CIO' CHE VERRA'
MEMORIZZATO.":FOR X=0 TO 1000:NEXT
13240 INPUT "POSIZIONARE IL NASTRO, QUI
NDI PREMERE RETURN:";Q$
13250 PRINT"
                                  ";
13260 OPEN 1,1,1,"ARTISTA":FOR I=0 TO 3:
PRINT#1,CO(I):NEXT:PRINT#1,LEN(D1$)
13270 FOR I=1 TO LEN(D1$):PRINT#1,ASC(MI
D$(D1$,I,1)) R$ ASC(MID$(D2$,I,1)):NEXT
13280 CLOSE 1:RETURN
```

Lo scopo di questo modulo è di permettere di definire sullo schermo un disegno di dimensioni ridotte, per poi salvarlo in modo efficiente.

Commenti

Linee 13030-13050: se nel modulo precedente è stato selezionato un MODO 6, seguito da R, queste tre linee permettono di accettare un ulteriore carattere, che deve essere 1 o 2. Se viene premuto 1, la posizione corrente del cursore viene definita come l'angolo superiore sinistro del rettangolo da salvare, 2 definisce l'angolo inferiore destro. In realtà queste due posizioni risultano esterne al disegno da

salvare, definiscono cioè una cornice esterna a ciò che va salvato. Le posizioni in memoria dei due angoli vengono immagazzinate nella matrice CO. Da notare il fatto che questa matrice non è stata dichiarata, dal momento che ha meno di 10 elementi — semplicemente l'introduzione di un valore la costruirà in modo soddisfacente. CO(0) o CO(2) viene posto al valore della posizione del cursore nella linea, indicato da PEEK(211). CO(1) oppure CO(3) viene posto a: numero di linea schermo + (posizione attuale in memoria — inizio area di schermo)/40.

Linee 13060-13280: permettono di salvare il rettangolo precedentemente definito.

Linea 13070: il colore di stampa viene temporaneamente eguagliato al colore del cursore.

Linea 13080: viene compiuto un controllo sul fatto di aver definito un rettangolo valido (cioè di lunghezza e larghezza determinate). In caso contrario, viene stampato un messaggio d'errore.

Linea 13110: i dati del disegno verranno temporaneamente memorizzati in una stringa, perciò si controlla se esiste la possibilità che questa sia troppo lunga. In questo caso viene stampato un messaggio d'errore.

Linee 13140-13160: facendo uso delle coordinate contenute nella matrice CO, sullo schermo vengono posti nuovamente (POKE) gli angoli del rettangolo e si domanda all'utente di confermare la correttezza del rettangolo da salvare.

Linea 13170: vengono inizializzate le due stringhe da utilizzare. I due cicli si combinano così per significare che verranno letti dallo schermo J caratteri (la larghezza del disegno) per I linee (l'altezza del disegno).

Linee 13180-13190: basandosi sugli indirizzi forniti dai cicli, vengono esplorate (PEEK) le memorie di schermo e di colore, e i valori trovati vengono sommati alla stringa di memorizzazione sotto forma di caratteri dotati del corrispondente valore di codice. Se stampate, le due stringhe così ottenute non avrebbero senso, si tratta semplicemente di un pratico metodo per memorizzare temporaneamente una serie di valori senza dover stabilire complessi puntatori a una posizione di una matrice. Fatto questo, una POKE forza un asterisco nella locazione di schermo e la sua funzione di colore viene posta sul nero, rendendo visibile l'elaborazione del disegno.

Linea 13200: si ottiene una copia di D1$ e D2$; vengono quindi utilizzati due altri cicli per riportare (POKE) sullo schermo i caratteri che erano stati memorizzati, contemporaneamente alle rispettive caratteristiche di colore. Ogniqualvolta un carattere viene posto nuovamente sullo schermo, le due stringhe vengono decurtate del carat-

tere più a sinistra, in modo da utilizzare sempre il primo carattere della stringa. È questo procedimento di accorciamento a richiedere la creazione di una copia temporanea delle due stringhe originali. L'unico scopo di questi due cicli è di rassicurare l'utente sul fatto che sta correttamente avendo luogo il salvataggio del disegno.

Linee 13240-13280: il disegno viene salvato su nastro.

Linea 13260: vengono salvati i valori di CO, insieme alla lunghezza di DI$ (che è uguale alla lunghezza di D2$).

Linea 13270: un ciclo, pari alla lunghezza di D1$, salva i valori dei caratteri di entrambe le stringhe (cioè i valori tratti dalle memorie di schermo e di colore). Sfortunatamente non è possibile salvare su nastro direttamente le stringhe stesse, dal momento che queste possono contenere caratteri non stampabili, che il 64 non è in grado di salvare sotto forma di stringa.

Collaudo del modulo 3.1.3

A questo punto dovreste poter salvare su nastro un disegno. Se la figura ricostruita risulta soddisfacente, è probabile che l'operazione sia stata condotta a termine in modo corretto, ma ciò si può controllare completamente solo dopo aver introdotto almeno il modulo principale del programma Parole, che utilizzerà i disegni in tal modo creati.

MODULO 3.1.4

```
14000 REM#******************************
14010 REM SALVATAGGIO SCHERMO
14020 REM*******************************
14030 INPUT"▒POSIZIONARE IL NASTRO, QUIN
DI PREMERE ▒RETURN■:";Q$
14040 PRINT"▒
           ";:OPEN1,1,1,"SCHERMO"
14050 FORI=0TO999:PRINT#1,PEEK(1024+I):P
RINT#1,PEEK(55296+I):NEXT:CLOSE1:RETURN
```

Se, nel corso dell'esecuzione del modulo 2, è stato predisposto il MODO 6 ed è stato premuto il tasto S, questo modulo assicurerà il salvataggio del contenuto di tutto lo schermo. Lo scopo viene raggiunto con il semplicissimo metodo di controllare i contenuti delle locazioni da 1024 a 2023 (memoria di schermo), più le corrispondenti locazioni di memoria del colore, e di salvarne i contenuti. Un successivo programma potrà leggere i valori dal nastro e forzarli (POKE) nelle stesse locazioni.

Riepilogo

Questo programma può risultare molto gratificante, ma il suo massimo contributo è la possibilità di costruire agevolmente complesse forme grafiche, modificarle a piacimento e richiamarle semplicemente per utilizzarle in seguito per altri programmi. Sulla base delle tecniche impiegate in questa sede, non dovreste avere difficoltà per ulteriori programmi di vostra progettazione, che richiedano d'intervenire con POKE sulle memorie di schermo e colore.

Ulteriori sviluppi

1) Non è stato previsto il salvataggio del colore di sfondo dello schermo — dovrebbe risultare un'aggiunta di semplice realizzazione.
2) Perché non inserire sulla linea di fondo dello schermo un'indicazione del modo impostato?

Artista: lista delle funzioni dei tasti

f1	Ridefinisce il colore del carattere sotto il cursore
f2	Ridefinisce il colore dello schermo
f3	Permette di modificare il colore di stampa
f4	Cancella il disegno corrente
f5	Attiva o disattiva l'inversione
f6	Modo di SAVE (salvataggio su nastro)
f7	Modo inattivo
f8	Permette di modificare il colore del cursore
←	Inserisce il secondo set di colori nei modi 1-3

Modo SAVE:

R	Seguito da 1 o 2 definisce gli angoli del rettangolo da salvare
D	Salva piccoli disegni
S	Salva tutto lo schermo

3.2 Caratteri

Indipendentemente dalla bontà del set di caratteri fornito da un micro, verrà il momento in cui non sarà disponibile il carattere desiderato. Può essere che vogliate scrivere in una lingua che richieda accenti particolari, o stampare astrusi simboli matematici, oppure che vi serva qualcosa di speciale per dare il tocco finale al vostro ultimo gioco. Qualsiasi sia la vostra necessità, il 64 non aspetta altro che di soddisfarla con le possibilità derivanti dai caratteri-utente.
All'accensione, tutti i possibili caratteri del 64 sono immagazzinati nella sua ROM — memoria di sola lettura — in una sezione che inizia all'indirizzo 53248. Ogni carattere è rappresentato da otto byte di memoria e la risultante griglia di 8 * 8 punti, che costituiscono ciascun carattere, corrisponde ai singoli bit degli otto byte. Se, per esempio, gli otto byte di memoria di un carattere valessero 128, 64, 32, 16, 8, 4, 2, 1, allora, in notazione binaria sarebbero 10000000, 01000000, 00100000, 00010000, 00001000, 00000100, 00000010, 00000001. Ponendo ora questi valori in una griglia, otteniamo:

```
10000000
01000000
00100000
00010000
00001000
00000100
00000010
00000001
```

I bit che sono a 1 (o accesi) definiscono un carattere (in questo caso una linea diagonale), con ogni bit attivo trasformato in un pixel dello schermo.

Va tutto bene, ma, dal momento che i dati del carattere sono memorizzati in una ROM, memoria di *sola lettura*, non possono essere modificati — sono predisposti in modo permanente all'atto della fabbricazione del chip. Fortunatamente, il 64 fornisce un buon modo per aggirare l'ostacolo, ma, per poterlo comprendere, dobbiamo prima esaminare il metodo impiegato per generare l'immagine video. Tutte le funzioni relative al video, nel 64, fanno capo a un chip separato, il Chip d'Interfaccia Video 6567 (o chip VIC II), che tratta sia lo schermo in sé che i caratteri da porre sullo schermo, definendo un'area di memoria in cui sarà immagazzinata l'informazione di schermo e un'altra area in cui scriverà i dati dei caratteri.
Diversamente da quanto ci si potrebbe aspettare, il VIC II non estrae i dati dei caratteri dalla ROM all'indirizzo 53248. La ragione è che il VIC II può vedere solo 16 K di memoria in ogni singolo istante, così, con la memoria di schermo nella sua posizione normale da 1024 a 2023 in memoria, i dati relativi ai caratteri devono essere estratti da qualche altra area di memoria tra 0 e 16383. Allo scopo, il sistema operativo imbroglia un poco e fa credere al VIC II che esista una copia dei dati del set di caratteri allocata da 4096 a 6143. Ogniqualvolta il VIC II legge in quell'area di memoria, trova i dati del set di caratteri, a dispetto del fatto che in realtà quell'area sarà probabilmente riempita da un programma BASIC.
La cosa può apparire un po' astrusa, ma è d'importanza fondamentale, dal momento che significa che, invece di cercare i dati relativi ai caratteri nella ROM, che non si può modificare, il VIC II li cerca nella RAM (memoria ad accesso casuale), cioè una memoria di lettura/scrittura che l'utente può raggiungere e alterare. Naturalmente non è proprio così semplice. Abbiamo già notato che, quando il VIC II guarda l'area di memoria da 4096 in avanti, non sono effettivamente quelli gli indirizzi che vede, ma un'immagine dei dati dei caratteri contenuti nella ROM. Fortunatamente, è questa una prerogativa di solo due dei blocchi di memoria all'interno del banco da 16K, 4096-6143 e 6144-8193. Se si istruisce il VIC II a ricercare i dati dei caratteri da uno qualsiasi degli altri blocchi da 2K contenuti in un banco da 16K, non vedrà l'immagine della ROM, ma prenderà i dati effettivamente in quel segmento di memoria, e li tratterà come se fossero il set di caratteri.
Ora la domanda è: quale blocco dobbiamo specificare? Il primo disponibile è 2048-4095, ma questo presenta il piccolo inconveniente di coincidere con l'inizio del programma BASIC, e forzarvi i nuovi dati dei caratteri sarà distruttivo nei confronti dei programma. Potremmo utilizzare i blocchi a 8192, 10240, 12288, 14336, ma sfortunatamente ciò significherebbe che dovremmo ridurre l'area disponibile per un programma BASIC in modo piuttosto drastico, perché altrimenti si correrebbe il rischio che un programma di dimensioni notevoli possa andare a sovrapporsi all'area usata per i caratteri. La soluzione qui adottata è di muovere ancora più in alto nella memoria l'intera area indirizzata dal VIC II.
Ricorderete che il VIC II è in grado di vedere uno spezzone di memoria da 16K per volta, tuttavia non importa quale blocco veda. Si hanno quattro blocchi del genere, con inizio a 0, 16384, 32768,

49152. Portandosi sul blocco a 49152, mentre da una parte si dà la massima area di memoria al BASIC, dall'altra sorge il problema che è là che si trova la ROM, così richiederemo al VIC II di indirizzare il blocco che inizia all'indirizzo 32768. Ciò fatto, rimane solo da specificare dove è il blocco da cui si prenderanno i dati dei caratteri e dove verranno allocati i dati di schermo. A questo punto ci saranno 30K di memoria disponibili per un programma BASIC (2048-32767) e la potenzialità per un set di caratteri-utente nella RAM sopra 32768. Senza dubbio tutto ciò sembra eccessivamente complesso. In effetti, data la versatile struttura di memoria del 64, è solo questione di alcune POKE, e il gioco è fatto. Sulla base dei cambiamenti operati da queste POKE, il programma seguente permetterà di ridefinire completamente o in parte il set di caratteri del 64 e di archiviarlo in modo che altri programmi possano raccoglierlo e utilizzarlo.

Caratteri: lista delle variabili

A$	Istruzione di una sola lettera ottenuta tramite GET
C1	Colore originale dello schermo alla locazione CC
CC	Posizione corrente del cursore nella memoria di schermo
CH	Numero del carattere corrente nel set di caratteri
CP	Puntatore alla locazione di memoria del carattere CH
MM	Valore introdotto per cambiare CP
P1	Riga della posizione del cursore sullo schermo
P2	Colonna della posizione del cursore sullo schermo
PP	Contenuto originale dello schermo alla locazione CC
TT%(7,7)	Matrice utilizzata per permettere la manipolazione dei dati nel carattere corrente

MODULO 3.2.1

```
11000 REM********************************
11010 REM RIDEFINIZIONE MEMORIA
11020 REM********************************
11030 POKE 53281,6:PRINT CHR$(142)
11040 POKE 52,128:POKE56,128
11050 POKE 56334,PEEK(56334)AND254
11060 POKE 1,PEEK(1)AND251
11070 FOR I=0 TO 2047:POKE 32768+I,PEEK(
53248+I):NEXT
11080 POKE 1,PEEK(1)OR4
11090 POKE 56334,PEEK(56334)OR1
11100 POKE 56578,PEEK(56578)OR3
11110 POKE 56576,(PEEK(56576)AND252)OR1
11120 POKE 648,136
11130 POKE 53272,32
```

Lo scopo di questo modulo è di portare a termine tutte le modifiche alla struttura di memoria precedentemente indicate e di copiare un set di caratteri iniziale nell'area di RAM specificata.

Commenti

Linea 11030: pone la macchina nel modo relativo ai caratteri maiuscoli, dato che solo il primo dei due set di caratteri disponibili sul

64 sarà utilizzabile, una volta che il VIC II non potrà più vedere l'immagine ROM.

Linea 11040: queste due POKE stabiliscono l'estremo superiore dell'area disponibile per il programma BASIC, in modo tale da impossibilitare qualsiasi interferenza dei programmi con l'area di memoria riservata ai caratteri. Al disotto di questa delimitazione sono disponibili 30K di memoria.

Linee 11050-11060: queste due POKE inibiscono la scansione della tastiera, in modo che nessun *interrupt* (interruzione) possa disturbare la successiva sezione del programma, e quindi rendono visibile al programma il set di caratteri della ROM disabilitando il normale processo d'ingresso/uscita. Durante il ciclo seguente, l'unico modo di fermare il programma sarà lo spegnimento della macchina.

Linea 11070: copia il set di caratteri dalla ROM all'area di memoria che inizia a 32768 — ciò implica il trasferimento di 2Kbyte.

Linee 11080-11090: riabilitano il normale regime di ingresso/uscita e ristabiliscono i normali interrupt.

Linee 11100-11110: queste due POKE prima preparano il VIC II a un trasferimento di blocco di memoria, e quindi specificano il blocco 1 (32768-49151).

Linea 11120: questa locazione è esterna al VIC II ed è la guida al sistema operativo per indicare dove va allocata la memoria di schermo — in questo caso a fare capo da 256 * 163 = 34816.

Linea 11130: la locazione alla quale il VIC II si aspetta di trovare sia i dati di schermo che quelli di carattere, all'interno del suo blocco di 16K, è imposta dai contenuti dell'indirizzo 53272 — i quattro bit più significativi per lo schermo, i rimanenti per il set di caratteri. Questa POKE pone i quattro bit più significativi a 0010, che significa il blocco da 1K che inizia da 32768 + 2048 per lo schermo, e quelli meno significativi a 0000, specificando che i dati di carattere saranno tratti da 32768 + 0. Per giungere ad altre possibili locazioni nel blocco da 16K, la formula per scrivere la POKE dovrà essere ((INIZIO SCHERMO — INIZIO BLOCCO)/1024) * 16 + ((CARATTERI — INIZIO BLOCCO)/2048). Lo schermo può iniziare solo a multipli interi di 1K all'interno del blocco mentre i caratteri possono iniziare solo a multipli interi di 2K (sui confini dei singoli segmenti da 1 e 2K). Notate che avremmo potuto lasciare lo schermo a 1024 e la memoria dei caratteri a 4096, solo che in questo blocco di 16K, come del resto nel blocco che inizia all'indirizzo zero, il VIC II vede un'immagine della ROM da 4096 in avanti.

Collaudo del modulo 3.2.1

Il controllo di questo modulo è piuttosto semplice. Mandatelo in esecuzione e la macchina si bloccherà per qualche momento — non c'è niente che possiate fare per intervenire.

Quando apparirà sullo schermo la scritta READY, non dovrebbe essere cambiato niente — cosa che indica che il modulo ha funzionato! In caso contrario, lo schermo si riempirà di segni incomprensibili.

MODULO 3.2.2

```
12000 REM********************************
12010 REM STAMPA GRIGLIA
12020 REM********************************
12030 CH=0:DIM TT%(7,7)
12040 PRINT "▒▒";:FOR I=1 TO 8:PRINT "▒▌
▌▌▌▌▌▌▌▌ ":NEXT
12050 PRINT "▒          "
12060 CP=32768+CH*8
12070 PRINT "▒▒";:FOR I=CP TO CP+7:FOR J
=7 TO 0 STEP-1
12080 IF (PEEK(I)AND 2↑J)=2↑J THEN PRINT
 "▒ ";
12090 IF (PEEK(I) AND 2↑J)=0 THEN PRINT
"■ ";
12100 NEXT J:PRINT:NEXT I
12110 PRINT "▒▒▒▒▒▌▌▌▌▌▌▌▌▌▌▌";"NUMERO DE
L CARATTERE:";CH;"▒▒▒▒▒▒▒"
12120 INPUT "NUM. DI SPOSTAMENTO PUNTATO
RE (0=RIDEF.)";MM:CH=CH+MM
12130 IF CH<0 THEN CH=0
12140 IF CH>255 THEN CH=255
12150 IF MM=0 THEN 13000
12160 GOTO 12040
```

Esistono molti metodi per introdurre nuovi caratteri nella relativa memoria. Potete, se lo desiderate, disegnarli su una matrice 8 * 8, tradurre in binario le linee di punti per poi convertirle in forma decimale, inserire i numeri sotto forma di istruzioni DATA e infine porre quest'ultime in memoria con delle POKE. Fortunatamente è molto più facile fare sì che sia il 64 a compiere il lavoro, disegnando la griglia del carattere corrente sullo schermo, per poi permetterne una agevole manipolazione.
Questo modulo disegna la griglia del carattere, il seguente rende possibile la manipolazione.

Commenti

Linee 12040-12050: viene tracciato un quadrato di 8 * 8 nell'angolo superiore sinistro dello schermo.

Linea 12060: si calcola la posizione dei dati del carattere corrente.

Linee 12070-12100: si provvede a stampare una versione ingrandita del carattere corrente nel quadrato tracciato. Notare l'uso di AND in questo contesto per estrarre il contenuto dei singoli bit all'interno degli otto byte di memoria del carattere. Tutto ciò che fa AND è la comparazione di due numeri binari per produrne un terzo che ha degli "1" solo nei bit che erano "1" in entrambi i numeri origina-

riamente confrontati. Così, se si esegue una AND tra 193 e 129 (in binario 11000001 AND 10000001), il risultato è 129 dal momento che il bit 6 del primo numero non è a "1" anche nel secondo. Eseguire una AND tra il valore di un byte e 2↑(J), dove J va da 0 a 7, evidenzierà il valore del bit J-esimo — ricordate che i bit sono numerati da 0 a 7, da destra a sinistra.

Linee 12110-12160: viene fornito il numero del carattere disegnato e l'utente ha la possibilità di muovere il puntatore su un altro dei 255 caratteri. Se si introduce il numero 0, il programma passa al modulo successivo.

Collaudo del modulo 3.2.2

Ancora una volta, il test è piuttosto semplice. Se il modulo è stato battuto in maniera corretta, l'esecuzione del programma porterà (dopo una breve pausa) al disegno sullo schermo di una versione ingrandita del carattere '@'. Dovreste inoltre riuscire a far scorrere gli altri caratteri.

MODULO 3.2.3

```
13000 REM#******************************
13010 REM RIDEFINIZIONE CARATTERI
13020 REM*******************************
13030 PRINT "
                 ":REM 40 SPAZI
13040 PRINT "'I' INVERSIONE",,,"'M' IM
MAGINE SPECULARE",,"'R' RETURN"
13050 PRINT "'1' COLORAZIONE PIXEL":PRI
NT "'0' CANCELLAZIONE PIXEL"
13060 PRINT "'T' ROTAZIONE",,,"'P' MEM
ORIZZAZIONE"
13070 PRINT"'D' SALVATAGGIO IN UN FILE
DATI","'C' CARICAMENTO SET DAL NASTRO"
13080 PRINT "'N' NORMALIZZAZIONE MEMORI
A E TERMINE"
13100 PRINT " TASTI CURSORE PER IL MOVI
MENTO"
13110 PRINT "";
13120 GET A$
13130 CC=PEEK(211)+PEEK(210)*256+PEEK(20
9):PP=PEEK(CC):C1=55296+CC-34816
13140 C2=PEEK(C1)
13150 POKE CC,42:POKE C1,1
13160 FOR I=1 TO 15:NEXT:POKE CC,PP:POKE
C1,C2:IF A$="" THEN 13120
13170 P1=INT((CC-34816)/40):P2=CC-(34816
+40*P1)
13180 IF (P1>0 AND A$="")OR(P1<7 AND A$
="") THEN PRINT A$;:GOTO 13120
13190 IF (P2>0 AND A$="")OR(P2<7 AND A$
="") THEN PRINT A$;:GOTO 13120
13200 IF A$="1" AND P1<8 AND P2<8 THEN P
RINT " ";
```

```
13210 IF A$="0" AND P1<8 AND P2<8 THEN P
RINT "▒█ ▢";
13220 IF A$<>"I" THEN 13280
13230 FOR I=0 TO 7:FOR J=0 TO 7
13240 IF PEEK(34816+I*40+J)=32 THEN 1326
0
13250 POKE 34816+40*I+J,32:POKE 55296+40
*I+J,182:GOTO 13270
13260 POKE 34816+I*40+J,160:POKE 55296+4
0*I+J,181
13270 NEXT J,I
13280 IF A$="R" THEN GOTO 12040
13290 IF A$<>"M" THEN 13370
13300 FOR I=0 TO 7:FOR J=0 TO 7:TT%(I,J)
=0:NEXT J,I
13310 FOR I=0 TO 7:FOR J=0 TO 7
13320 IF PEEK(34816+40*I+J)=160 THEN TT%
(I,J)=1
13330 NEXT J,I
13340 PRINT"▒";:FORI=0TO7:FORJ=7TO0 STEP
-1:IF TT%(I,J)=1 THEN PRINT "█▒ █";
13350 IF TT%(I,J)=0 THEN PRINT "▒ ";
13360 NEXT J:PRINT:NEXT I
13370 IF A$<>"T" THEN 13450
13380 FOR I=0 TO 7:FOR J=0 TO 7:TT%(I,J)
=0:NEXT J,I
13390 FOR I=0 TO 7:FOR J=0 TO 7
13400 IF PEEK(34816+40*I+J)=160 THEN TT%
(7-J,7-I)=1
13410 NEXT J,I
13420 PRINT "▒";:FORI=0TO7:FORJ=7TO0 STE
P-1:IF TT%(I,J)=1 THEN PRINT "█▒ █";
13430 IF TT%(I,J)=0 THEN PRINT "▒ ";
13440 NEXTJ:PRINT:NEXTI
13450 IF A$<>"P" THEN 13510
13460 FOR I=0 TO7:TT%(0,I)=0:NEXT
13470 FOR I=0 TO7:FOR J=0 TO 7
13480 IF PEEK(34816+40*I+J)=160 THEN TT%
(0,I)=TT%(0,I)OR 2↑(7-J)
13490 NEXT J,I
13500 FOR I=0 TO 7:POKE CP+I,TT%(0,I):NE
XT:GOTO 12040
13510 IF A$<>"D" THEN 13550
13520 OPEN 1,1,1,"CARATTERI"
13530 FOR I=0 TO 2047:T%=PEEK(32768+I):P
RINT#1,T%:NEXT
13540 CLOSE 1
13550 IF A$<>"C" THEN 13590
13560 OPEN 1,1,0,"CARATTERI"
```

```
13570 FOR I=0 TO 2047:INPUT#1,T:POKE 327
68+I,T:NEXT
13580 CLOSE 1
13590 IF A$<>"N" THEN 13660
13600 POKE 52,160:POKE 56,160:CLR
13610 POKE 56578,PEEK(56578)OR3
13620 POKE 56576,(PEEK(56576)AND252)OR3
13630 POKE 53272,21
13640 POKE 648,4
13650 END
13660 GOTO 13120
```

Questo modulo esegue svariate funzioni relative alla manipolazione di un carattere sullo schermo, permettendone la ridefinizione, il riposizionamento in memoria e il salvataggio su nastro, tra le altre cose.

Commenti

Linee 13030-13100: stampa di concise istruzioni per l'uso del modulo.

Linee 13120-13160: questo modulo non riserva sorprese. Si tratta semplicemente della routine a cursore lampeggiante presa dal programma Artista.

Linea 13170: posizione del cursore: P1 è la riga partendo dall'alto, P2 è la colonna contata da sinistra.

Linee 13180-13190: limiti di spostamento del cursore con il quadrato 8 * 8.

Linee 13200-13210: premendo 1 si colora di verde un quadratino, premendo 0 se ne cancella uno. Vengono ignorate le istruzioni di stampa che porterebbero il cursore fuori dalla griglia definita.

Linee 13230-13270: questi due cicli scandiscono il quadrato invertendo gli elementi colorati e non, e producono così un carattere inverso.

Linea 13280: premendo R si torna al modulo precedente.

Linee 13290-13360: questa routine produce un'immagine speculare di una qualsiasi combinazione di punti — cioè il carattere appare come visto da dietro.

Linea 13300: si azzera la matrice TT%.

Linee 13310-13330: i contenuti dello schermo vengono trasferiti nella matrice. Non è possibile elaborare direttamente lo schermo, dal momento che ciò potrebbe provocare il trasferimento del quadrato da sinistra a destra e di seguito una doppia lettura dello stesso, cosa che porterebbe a un risultato privo di significato.

Linee 13340-13360: trasferito il contenuto della griglia dei punti alla

matrice, l'informazione viene riletta sullo schermo, ma l'elemento orizzontale viene rovesciato, in modo che la posizione 7 viene a trovarsi nella posizione 0.

Linee 13370-13440: i contenuti della matrice di punti vengono ruotati di 90 gradi in senso antiorario.

Linee 13420-13440: i contenuti della matrice TT% vengono nuovamente portati sullo schermo, ma ruotati in senso antiorario — dunque la posizione 0,7 diviene la posizione 0,0 e la 0,0 diventa la 7,0.

Linee 13450-13500: il carattere ridefinito viene riportato nella memoria dei caratteri. Diventa ora una parte permanente del set definito dall'utente.

Linea 13460: dal momento che per un carattere si richiedono solo otto byte, è necessario cancellare solo otto byte della matrice, la linea zero, da 0 a 7.

Linee 13470-13490: viene esplorata ogni linea della matrice e, quando si trova un pixel colorato, questo viene tradotto in un singolo bit di uno degli otto byte utilizzati per definire il carattere. Dopo avere utilizzato degli operatori AND per leggere i singoli bit, notate l'uso di OR per manipolarli. Eseguendo l'operazione OR tra due numeri binari, tutti i bit che si trovano a "1" in uno (o entrambi) degli operandi, sono a "1" nel risultato. Dunque calcolare l'OR di un numero con 2 ↑ (J), dove J va da 0 a 7, significa porre a "1" il J-esimo bit, indipendentemente dalla sua condizione primitiva.

Linea 13500: gli otto byte della matrice vengono posti in memoria nella posizione precedentemente occupata dal carattere ridefinito.

Linee 13510-13540: l'area di memoria da 32768 in avanti viene salvata sul nastro sotto forma di numeri interi.

Linee 13550-13560: è possibile caricare dal nastro un set di caratteri precedentemente memorizzato per ulteriori manipolazioni. NB: questo è un esempio di come estrarre un nuovo set di caratteri da un altro programma, per un successivo utilizzo.

Linee 13590-13650: se il programma è concluso, la memoria deve essere riposta nelle sue condizioni originali — a meno che non vogliate continuare a usare il nuovo set di caratteri con un altro programma che state per caricare. Dimenticare di riposizionare la memoria significherebbe privare i programmi seguenti di 8K di memoria e forzarli a usare il nuovo set di caratteri.

Linea 13600: si restituisce al BASIC tutta la sua potenziale estensione.

Linee 13610-13620: si ripristina il banco di memoria indirizzato dal VIC II (3, cioè 0-16383).

Linee 13630-13640: fanno sì che lo schermo inizi di nuovo a 1024 (sua posizione naturale) e la memoria dei caratteri è riportata da 4096 in avanti.

Collaudo del modulo 3.2.3

Dato che questo è un modulo piuttosto lungo, con svariate funzioni, si suggerisce di collaudare separatamente ogni funzione dopo averla introdotta. Notate che, se una funzione particolare risulta difettosa e avete inserito delle modifiche a una linea, non è necessario eseguire tutto il programma dal principio. Battete semplicemente GOTO 12000, dato che il set di caratteri, che si trova oltre l'area BASIC, e la struttura della memoria non vengono disturbati dall'introduzione di nuove linee. Se tutto funziona come dovrebbe, saranno disponibili le funzioni descritte nei commenti.

Riepilogo

Come non mancherete di notare, si tratta di un programma estremamente piacevole da usare anche di per se stesso, ma tutte le sue possibilità si rivelano pienamente in ciò che può fare nel ravvivare l'uscita di altri vostri programmi. Dal momento che in realtà non riloca il BASIC, ma ne limita solo lo spazio disponibile, si possono caricare in macchina altri programmi che facciano uso del set di caratteri ridefinito. Se è stata spenta la macchina, o rinormalizzata la memoria, dopo aver ridefinito il set di caratteri, tutto ciò che bisogna fare è aggiungere il primo modulo (eccetto le linee 11050-11090) all'inizio dei programmi che seguono, per poi ricaricare dal nastro il set di caratteri ridefinito tramite la routine a 13560-13580. Tuttavia, ricordate che, se ridefinite "A" come il carattere "invasore spaziale", allora ogni "A" stampata dal programma, persino nel listato, sarà quella ridefinita. Per salvaguardare la leggibilità è solitamente meglio ridefinire i caratteri grafici!
Questo programma, al di là del suo impiego pratico, rappresenta una valida introduzione ad alcune delle possibilità dischiuse dalla versatile struttura di memoria del 64 e alle tecniche necessarie per ottenere il massimo da quanto disponibile. Volendo approfondire le tecniche di manipolazione della memoria, è necessario procurarsi una copia della *Guida di Riferimento per il Programmatore* — dopo questo programma non dovreste avere alcuna difficoltà nel comprendere e applicare ciò che vi troverete.

Ulteriori sviluppi

1) Una semplice aggiunta al programma potrebbe essere una routine per scambiare tra loro le posizioni di due caratteri, oppure per porre un carattere ridefinito in un'altra locazione del set.
2) Costruire un intero set di caratteri con questo programma potrebbe diventare una cosa estremamente lunga. Perché non provare ad aggiungere qualche funzione di manipolazione di blocchi che permetta d'invertire, ruotare, riprodurre specularmente, ecc. un insieme di vari caratteri compresi tra limiti da specificare? I listati appaiono estremamente interessanti con tutte le lettere rovesciate!

3.3 Animazioni

Introdotto il programma Caratteri, abbiamo spianato la via per l'esame di una delle caratteristiche del 64 che altri possessori di microcomputer possono solamente sognare: le animazioni. Con l'avvento del 64, sono definitivamente finiti i tempi in cui solo i programmatori in linguaggio macchina potevano fare muovere a loro piacimento disegni ad alta risoluzione per tutto lo schermo, con impressionante realismo. Specialmente nel campo dei giochi, le animazioni rappresentano una rivoluzione tra i micro di prezzo contenuto.
In sostanza, un'animazione differisce molto poco dai caratteri-utente con i quali abbiamo già lavorato. La creazione della "funzione" animazione ha richiesto notevoli doti d'immaginazione e competenza tecnica, ma per quanto riguarda l'utente, un'animazione è solamente un carattere un po' più grande, che si può muovere sullo schermo in modo più versatile.
Come i caratteri del normale set di caratteri, le animazioni sono definite da una serie di byte memorizzati in RAM. Tuttavia, invece di una griglia 8 * 8, le animazioni usano una matrice di 24 punti in orizzontale per 21 in verticale. Chiaramente questa griglia non si può definire tramite i 64 punti presenti in 8 byte. Infatti, ogni riga è definita da tre byte (24 bit) e, dato che ci sono 21 righe, sono necessari 63 byte per definire un'animazione o *sprite*.
I dati delle animazioni si possono memorizzare in qualsiasi zona sicura all'interno del blocco di memoria di 16K indirizzato dal VIC II. All'interno di questo blocco si possono definire fino a otto animazioni contemporaneamente, ma si possono, se necessario, tenere di riserva molti più disegni, per renderli istantaneamente attivi.
Le principali locazioni di memoria che controllano l'uso delle animazioni sono le seguenti:
a) 2040-2047: queste otto locazioni sono i puntatori all'animazione. Hanno la funzione di indicare da dove ricavare i dati per ogni singola animazione nel blocco da 16K. Dal momento che le animazioni sono memorizzate in blocchi di 64 byte (benché ne usino solo 63), i 256 valori che si possono forzare in ogni puntatore permettono di coprire l'intero blocco di 16K. Così i dati dell'animazione 2 verranno estratti dalla memoria a 64 * PEEK(2042).
b) 53269: registro di abilitazione animazione. Un'animazione è visibile solo quando è attivato il bit corrispondente di questo registro.
c) 53248-53264: registri posizione. Questi lavorano in coppie da 53248 a 53263, definendo le coordinate X e Y sullo schermo dell'angolo superiore sinistro della griglia dell'animazione. Tuttavia, dato che in realtà lo schermo è più ampio del valore di 255, massimo memorizzabile in un singolo byte, e raggiunge i 320 pixel, si utilizza un bit della locazione 53264 per segnare se la posizione di ogni oggetto sull'asse X sia maggiore di 255. Tutto ciò porta a un totale di 512 posizioni possibili sull'asse X e 256 sull'asse Y.
d) 53287-53294: registri colore dell'animazione. Ogni oggetto può assumere uno qualsiasi dei 16 colori del 64, semplicemente forzando il giusto valore nel registro appropriato.
In realtà ci sono altre locazioni di rilievo, ma queste basteranno per poter proseguire.

La questione finale da decidere è dove porre i dati degli sprite da animare. Se volete solo tre animazioni, un luogo comodo è il buffer d'ingresso/uscita per le cassette, che si trova da 828 a 1019 (ovviamente non si possono caricare o salvare dati finché le animazioni sono allocate in quella posizione). Se volete un numero maggiore di oggetti, sarà necessario separare un'area di memoria per questi: esattamente la stessa situazione dei caratteri-utente. Per amore di varietà, nel nostro programma di definizione di animazioni, adotteremo una soluzione differente da quella offerta nel programma Caratteri. Ciò che faremo è spostare l'inizio del programma BASIC da 2048 a 4096, lasciando così 2K di memoria dove porre fino a 32 insiemi distinti di dati relativi alle figure da disegnare. Ciò risulta conveniente per il fatto di non comportare assolutamente spostamenti della struttura della memoria video — comporta, invece, il riposizionamento dell'indirizzo iniziale del BASIC, prima di caricare il programma.
Ciò fatto, il programma, come il generatore di caratteri, permetterà la definizione e la manipolazione delle matrici di punti e l'opzione di salvataggio su nastro per usi successivi. Il modo più semplice per introdurre questo programma è quello di caricare prima Caratteri, e poi adattare quel programma.

Programma di caricamento per Animazioni

Le linee che seguono *non* fanno parte del programma principale, sono pensate per essere inserite nel 64 e salvate sul nastro prima di scrivere e salvare il programma principale. La funzione è di riposizionare l'inizio del BASIC, per poi caricare il programma principale nella memoria riconfigurata.

```
100 REM********************************
110 REM CARICAMENTO
120 REM********************************
130 POKE43,1:POKE44,16:POKE 4096,0:CLR
140 LOAD"ANIMAZIONI"
```

Commenti

Le locazioni 43 e 44 sono i puntatori di sistema verso l'inizio del programma BASIC, normalmente contenenti i valori 1 e 8 (locazione 1 + 256 * 8 = 2049). Tutto ciò che fa la linea principale è modificare questo valore, portandolo a 4097. Il primo byte del programma deve sempre essere 0, valore che viene quindi imposto dalla POKE; viene poi cancellata la memoria, completandone la riconfigurazione. Fatto ciò, viene automaticamente caricato il programma principale. A costo di annoiarvi, ricordate che *non* deve far parte del programma principale — altrimenti verrebbero tagliati i primi 2K di programma non appena lanciata l'esecuzione.

Animazioni: lista delle variabili

SP	Indirizzo del puntatore all'animazione corrente
SC	Indirizzo del registro colore dell'animazione
SS	Inizio dei dati dell'animazione
FNS(SN)	Inizio del blocco di dati dell'animazione SN
SN	Numero dell'animazione

TT%(20, 23)	Matrice per la manipolazione temporanea dei dati dell'animazione
MM	Valore per il movimento di SN

(Si veda anche il programma Caratteri).

MODULO 3.3.1

```
12000 REM#*****************/*****************
12010 REM COSTRUZIONE PUNTATORE ANIMAZ.
12020 REM**********************************
12030 SP=2040:SE=53269:SS=2048
12040 DEF FNS(SN)=SS+64*(SN)
12050 SN=0:DIM TT%(20,23):POKE53281,6
```

Le variabili qui dichiarate sono spiegate nella lista delle variabili.

MODULO 3.3.2

```
13000 REM#*********************************
13010 REM STAMPA GRIGLIA
13020 REM**********************************
13030 POKE53269,1:POKE53287,1:POKE SP,FN
S(SN)/64
13040 POKE53248,0:POKE53264,1:POKE53249,
80
13050 PRINT "";:FOR I=1 TO 11:PRINT "
                        "
13060 PRINT ""
:NEXT
13070 PRINT "
  "
13080 PRINT "
"
13090 PRINT "";:FOR I=FNS(SN) TO FNS(S
N)+62 STEP 3:FOR J=0 TO 2
13100 FOR K=7 TO 0 STEP-1
13110 IF (PEEK(J+I)AND 2↑K)=2↑K THEN PRI
NT "●";
13120 IF (PEEK(I+J) AND 2↑K)=0 THEN PRIN
T " ";
13130 NEXT K,J:PRINT:NEXT I
13140 PRINT "NUMERO DEL DISEGNO:";SN
13150 INPUT "SPIAZZAMENTO PUNTATORE (0=R
IDEF.)";MM:SN=SN+MM
13160 IF SN<0 THEN SN=0
13170 IF SN>31 THEN SN=31
13180 IF MM=0 THEN POKE 53248,0:POKE 532
64,1:POKE53249,200:GOTO 14000
13190 GOTO 13000
```

Quasi uguale al modulo per il tracciamento della griglia nel programma Caratteri.

Commenti

Linea 13030: 53269 è il registro di abilitazione delle animazioni — questa POKE attiva il bit zero e l'animazione corrispondente. 53287 è il registro colore dell'animazione zero e la POKE pone il colore sul bianco. Il puntatore all'animazione zero viene fatto puntare al primo blocco di 64 byte dell'area di memoria riservata.

Linea 13040: l'animazione zero è posta a 256 sull'asse X e a 80 sull'asse Y.

Linee 13050-13080: viene stampato il profilo della griglia.

Linee 13090-13130: il ciclo I controlla i dati dell'animazione in gruppi di tre, il ciclo J controlla ogni byte, il K ogni bit. Viene stampato un cerchio per ogni bit attivo.

Linee 13140-13190: l'utente può spostare il puntatore all'animazione. Notate che si modifica l'area di memoria puntata dal puntatore, non l'effettivo puntatore. In questo programma useremo sempre l'animazione zero.

Collaudo del modulo 3.3.2

Facendo girare questo modulo (ricordate di lanciare prima il programma per il caricamento) apparirà una figura "sporca" alla destra della griglia e singoli punti di quest'ultima verranno riempiti. Alle volte risulta difficoltoso vedere la corrispondenza tra le due figure a causa delle ombre automaticamente poste nelle animazioni. Due bit attivi sulla stessa linea, con uno spazio interposto, appariranno in realtà come un singolo blocco nero.

MODULO 3.3.3

```
14000 REM#******************************
14010 REM RIDEFINIZIONE ANIMAZIONI
14020 REM*******************************
14030 PRINT "
                ":REM 39 SPAZI
14040 F$="                             "
14050 PRINTF$;"'I' INVERS"
14060 PRINTF$;"'M' IMMAGINE"
14070 PRINTF$;"  SPECULARE"
14080 PRINTF$;"'R' RETURN"
14090 PRINTF$;"'1' ATTIVA"
14100 PRINTF$;"'0' CANCELLA"
14110 PRINTF$;"'T' RUOTA"
14120 PRINTF$;"'P' MEMORIZZ"
14130 PRINTF$;"'S' SALVA"
14140 PRINTF$;"'L' CARICA"
14150 PRINTF$;"'E' TERMINA"
14160 PRINTF$;"'X' SCAMBIA"
14170 PRINTF$;"'C' COLORE"
14180 PRINTF$;"TASTI CURSORE"
14190 PRINTF$;"  PER MUOVERE"
14200 PRINT"";
```

```
14210 GETA$
14220 CC=PEEK(211)+PEEK(210)*256+PEEK(20
9):PP=PEEK(CC):C1=55296+CC-1024
14230 C2=PEEK(C1)
14240 POKE CC,42:POKE C1,1
14250 FOR I=0TO 15:NEXT:POKE CC,PP:POKEC
1,C2:IF A$="" THEN 14210
14260 P1=INT((CC-1024)/40):P2=CC-(1024+4
0*P1)
14270 IF (P1>0 AND A$="[illegible]") OR (P1<20 AND
 A$="[illegible]") THEN PRINT A$;:GOTO 14210
14280 IF (P2>0 AND A$="[illegible]") OR (P2<23 AND
 A$="[illegible]") THEN PRINT A$;:GOTO 14210
14290 IF A$="1" THEN PRINT "[illegible]";:GOTO 1
4210
14300 IF A$="0" THEN PRINT "[illegible]";:GOTO
 14210
14310 IF A$<>"I" THEN 14370
14320 FOR I=0 TO 20:FOR J=0 TO 23
14330 IF PEEK(1024+I*40+J)=32 THEN 14350
14340 POKE 1024+I*40+J,32:POKE 55296+40*
I+J,182:GOTO 14360
14350 POKE 1024+40*I+J,81:POKE 55296+40*
I+J,181
14360 NEXT J,I:GOTO 14210
14370 IF A$="R" THEN GOTO13030
14380 IF A$<>"M" THEN 14460
14390 FOR I=0 TO 20:FOR J=0 TO 23:TT%(I,
J)=0:NEXT J,I
14400 FOR I=0 TO 20:FOR J=0 TO 23
14410 IF PEEK(1024+I*40+J)=81 THEN TT%(I
,J)=1
14420 NEXT J,I
14430 PRINT"[illegible]";:FORI=0TO20:FORJ=23TO0 ST
EP-1:IF TT%(I,J)=1 THEN PRINT "[illegible]";
14440 IF TT%(I,J)=0 THEN PRINT "[illegible] ";
14450 NEXT J:PRINT:NEXT I:GOTO 14210
14460 IF A$<>"T" THEN 14540
14470 FOR I=0 TO 20:FOR J=0 TO 23:TT%(I,
J)=0:NEXT J,I
14480 FOR I=0 TO 20:FOR J=0 TO 20
14490 IF PEEK(1024+40*I+J)=81 THEN TT%(2
0-J,20-I)=1
14500 NEXT J,I
14510 PRINT"[illegible]";:FORI=0TO20:FORJ=20TO0 ST
EP-1:IF TT%(I,J)=1 THEN PRINT "[illegible]";
14520 IF TT%(I,J)=0 THEN PRINT "[illegible] [illegible]";
14530 NEXT J:PRINT:NEXT I:GOTO 14210
14540 IF A$<>"P" THEN 14600
```

```
14550 FOR I=0 TO 20:FOR J=0 TO 2:TT%(I,J
)=0:NEXT J,I
14560 FOR I=0 TO 20:FOR J=0 TO 2:FOR K=0
 TO 7
14570 IF PEEK(1024+40*I+8*J+K)=81 THEN T
T%(I,J)=TT%(I,J) OR 2↑(7-K)
14580 NEXT K,J,I
14590 FOR I=0 TO 20:FORJ=0TO2:POKEFNS(SN
)+3*I+J,TT%(I,J):NEXT J,I:GOTO 13030
14600 IF A$<>"S" THEN 14660
14610 INPUT "[illegible]NUM
ERO DI ANIMAZIONI DA SALVARE";NN
14620 IF NN<1 OR NN>32 THEN 14610
14630 OPEN 1,1,1,"ANIMAZIONI":PRINT#1,NN
14640 FOR I=0 TO NN*64-1:T%=PEEK(SS+I):P
RINT#1,T%:NEXT
14650 CLOSE1
14660 IF A$<>"L" THEN 14700
14670 OPEN 1,1,0,"ANIMAZIONI":INPUT#1,NN
14680 FOR I=0 TO NN*64-1:INPUT#1,T:POKE
SS+I,T:NEXT
14690 CLOSE 1
14700 IF A$<>"E" THEN 14720
14710 POKE53269,0:POKE 43,1:POKE44,8:POK
E 2048,0:CLR:END
14720 IF A$<>"X" THEN 14780
14730 INPUT"[illegible]NUM
ERO DA SCAMBIARE:";S2
14740 IF S2<0 OR S2>31 THEN 14730
14750 FOR I=0 TO 62:LET T1=PEEK(I+FNS(SN
)):POKEI+FNS(SN),PEEK(I+FNS(S2))
14760 POKE I+FNS(S2),T1:NEXT
14770 GOTO 13000
14780 IF A$<>"C" THEN 14880
14790 IF PEEK(53276)AND1=1 THEN POKE 532
76,0:GOTO 14210
14800 INPUT "[illegible]DEF
INIRE IL COLORE PER 01 (0-15):";C1
14810 IFC1<0ORC1>15THENPRINT"]
                               ":GOTO 14800
14820 INPUT "[illegible]DEF
INIRE IL COLORE PER 10 (0-15):";C2
14830 IFC2<0ORC2>15THENPRINT"]
                               ":GOTO 14820
14840 INPUT "[illegible]DEF
INIRE IL COLORE PER 11 (0-15):";C3
14850 IFC3<0ORC3>15THENPRINT"]
                               ":GOTO 14840
14860 POKE53285,(PEEK(53285)AND240)ORC1:
POKE53287,(PEEK(53287)AND240)ORC3
```

```
14870 POKE 53286,(PEEK(53286)AND240)ORC2
:POKE53276,1:GOTO 14210
14880 GOTO 14210
14890 END
```

Questo modulo ha la stessa funzione del modulo di ridefinizione caratteri del programma precedente.

Commenti

Linee 14050-14190: istruzioni per l'uso del modulo.

Linee 14210-14250: modulo standard per lo spostamento del cursore.

Linea 14260: riga e colonna del cursore sulla matrice dei punti.

Linea 14370: R ritorna al modulo precedente.

Linee 14460-14530: rotazione oraria.

Linee 14540-14590: l'animazione ridefinita viene nuovamente posta in memoria. Il ciclo I scandisce ogni riga della griglia, il ciclo J si muove entro gruppi di tre byte, il K seleziona ogni bit.

Linee 14600-14650: i dati vengono salvati su nastro. L'utente ha la facoltà di dichiarare quante figure bisogna salvare. Ciò rende possibile il salvataggio di tre animazioni che si possono memorizzare nel buffer d'ingresso del nastro di un programma seguente.
Se viene richiamato dal nastro uno sprite per ulteriori modifiche, vengono perse le animazioni attualmente in memoria.

Linee 14700-14710: questa routine disabilita lo sprite, normalizza la memoria e termina il programma.

Linee 14720-14770: X permette di scambiare i dati dell'animazione con quelli di un'altra posizione — particolarmente utile alla costruzione di una terna.

Linee 14780-14870: questa routine permette di entrare o uscire dal modo multi-colore.

Linea 14790: se è attivo il modo multi-colore (cioè è a "1" il bit corrispondente nel registro degli sprite multi-colore a 53276) e si richiama questa funzione, il modo multi-colore viene disattivato per lo sprite zero.

Linee 14800-14840: gli enigmatici 01, 10 e 11 di questi messaggi si riferiscono alle combinazioni di bit sulla griglia dell'animazione. Trovandosi nel modo multi-colore, questa viene considerata come se avesse solo 12 punti in senso orizzontale (benché punti di lunghezza doppia). I bit vengono letti in coppie partendo da sinistra e, naturalmente, formano coppie di 00, 01, 10 o 11. Ognuna di queste combinazioni produrrà un colore diverso nel modo multi-colore, dove

00 è lo sfondo dello schermo colorato dal registro del multi-colore a 53285. Il colore 10 è governato dal normale registro di colore dell'animazione. Il colore 11 proviene dal registro del multi-colore posto a 53286.

Collaudo del modulo 3.3.3

Una volta battuto il modulo, dovrebbero divenire utilizzabili tutte le funzioni descritte tra i commenti. Come nel caso del modulo equivalente di Caratteri, è meglio controllare ogni funzione non appena introdotta.

Riepilogo

Qualche riflessione su questo programma mostrerà come sia facile utilizzare le animazioni, una volta comprese le funzioni di alcune locazioni di memoria. Il programma stesso fornirà una serie infinita di animazioni che si possono salvare su un nastro separato per usi successivi. Le tecniche contenute in questo programma renderanno cosa semplice l'uso ottimale di simili animazioni nelle vostre creazioni.

Ulteriori sviluppi

1) Il programma non prevede un'altra possibilità legata alle animazioni, cioè la funzione di espansione, che raddoppia l'altezza o la larghezza dell'oggetto (stesso numero di byte, solo allungati). È una funzione facilissima da aggiungere, dato che tutto ciò che comporta è l'attivazione del bit corrispondente nel registro 53277 per l'espansione orizzontale e nel 53271 per quella verticale. Agli scopi di questo programma, il valore corretto è zero.
2) Sarebbe utile poter caricare dal nastro solo parte di un insieme di animazioni, per esempio una alla volta, per decidere se desiderate introdurla nell'insieme corrente. Una piccola modifica alla routine di caricamento dovrebbe permettervi di ottenere questo risultato.

3.4 Alta risoluzione

Benché le possibilità fornite dai caratteri-utente e dalle animazioni siano pressoché illimitate, il 64 mette a disposizione un ulteriore modo grafico fondamentale, la grafica in *bit-map*. Ciò significa che, invece di essere in grado di indirizzare un minimo di una delle 1000 posizioni di carattere dello schermo normale, l'utente ha la possibilità di agire su ogni singolo pixel (forma contratta di *picture element* = elemento d'immagine) o punto sullo schermo. In questo *modo* si possono tracciare sullo schermo curve e segni lineari, anche se per sfruttarlo completamente dovrete entrare in possesso della cartuccia di espansione grafica del 64, che vi metterà a disposizione svariati versatili comandi grafici.
Per comprendere il programma qui presentato, è necessario saperne di più su come è costruito lo schermo nel modo bit-map. Lo schermo in sé contiene 320 * 200 posizioni distinte, per un totale di 64000. Per poterle memorizzare separatamente, sono necessari 8000 byte di memoria, pari a 64000 bit distinti. Ciascuna delle normali posizioni dei caratteri necessita di 8 byte (la griglia di 8 * 8 che abbiamo usato per la grafica-utente). Iniziando dall'angolo superiore sinistro dello schermo, i primi 8 byte (da 0 a 7) della memoria di schermo vengono utilizzati per creare quella che sullo schermo ordinario sa-

rebbe la prima posizione di carattere. I secondi 8 byte formano la seconda matrice di 8 * 8 punti, e così via per tutta la linea. Dal momento che in ogni linea sono presenti 40 posizioni di carattere, ogni linea richiede 320 byte. Dato che il modo bit-map permette l'indirizzamento dei singoli pixel, questa linea di matrici 8 * 8 è in grado di contenere otto linee dello spessore di un singolo pixel (benché tracciate tutte insieme appaiano come una barra compatta).
La memoria di 8K necessaria per contenere lo schermo nel modo bit-map è ovviamente impossibile da contenere nella normale memoria di schermo da 1K, né, in pratica, è possibile utilizzare quell'area come sua parte, dato che le locazioni da 1024 a 2023 sono utilizzate per contenere le informazioni di colore dello schermo in bit-map. La soluzione adottata nel programma che segue è di allocare lo schermo a far conto da 8192, lasciando 6K di memoria per il programma BASIC, con l'opzione di rilocarlo, nel caso di sue estensioni o sviluppi. Con il programma presentato in queste pagine, potrete utilizzare lo schermo come una tavola da disegno, usando, al posto della matita, le frecce di movimento del cursore oppure un semplice algoritmo di tracciamento linee.

Alta risoluzione: lista delle variabili

DX	Distanza lungo l'asse X tra gli estremi di una linea
DY	Distanza lungo l'asse Y tra gli estremi di una linea
FN PE	Valore da forzare in PP per cancellare il pixel X, Y
FN PP	Locazione del byte in cui cade il pixel X, Y
FN PV	Valore da forzare in PP per attivare il pixel X, Y
MO	*Modo* corrente del programma
SC	Inizio dello schermo
SL	Inclinazione della linea da tracciare
X1, X2	Coordinate X degli estremi della linea
Y1, Y2	Coordinate Y degli estremi della linea

MODULO 3.4.1

```
10000 REM#*******************************
10010 REM INIZIALIZZA ALTA RISOLUZIONE
10020 REM********************************
10022 CL$="":INPUT "♥PULIZIA SCHERMO (S
/N):";CL$
10025 REM POKE 44,64:POKE 43,1:POKE 1638
4,0:CLR
10027 DEF FNPP(X)=SC+320*INT(Y/8)+8*INT(
X/8)+(Y AND 7)
10028 DEF FNPV(X)=PEEK(FNPP(X)) OR (2↑(7
-(X AND 7)))
10029 DEF FNPE(X)=PEEK(FNPP(X)) AND (255
-2↑(7-(X AND 7)))
10030 POKE 53272,(PEEK(53272))OR 8:POKE
53265,PEEK(53265) OR 32:SC=8192
10035 IF CL$="N" THEN 10050
10040 FOR I=SC TO SC+7999:POKE I,0:NEXT
10050 FOR I=1024 TO 2023:POKE I,6*16+12:
NEXT
10060 MO%(0)=2:MO%(1)=5:MO%(2)=10
```

Questo modulo configura la memoria di schermo per il modo bit-map, definisce alcune utili funzioni e pulisce lo schermo in alta risoluzione.

Commenti

Linea 10025: le POKE di questa frase REM non sono necessarie all'esecuzione di questo programma. Sono state incluse allo scopo di fornire le informazioni necessarie per rilocare il (programma) BASIC, nel caso voleste sviluppare il programma tanto da rendere possibile che si sovrapponga alla memoria di schermo da 8192 in avanti. Come nel caso del programma Animazioni, le POKE vanno poste in un programma di caricamento da lanciare *prima* del programma principale. In questa versione il programma funziona perfettamente all'interno della memoria di 6K fino a 8192 — non è nemmeno necessario porre un limite superiore al BASIC.

Linee 10027-10029: lo scopo di queste istruzioni è spiegato nella lista delle variabili.

Linea 10030: 53272 è il registro normalmente usato per controllare la posizione in cui il VIC II cerca i dati dei caratteri; in questo caso imporrà l'inizio dello schermo in bit-map. Forzando 8 in questa locazione, si pone l'inizio dello schermo a 8192. Forzando 32 in 53265, si attiva il modo bit-map.

Linee 10035-10040: alla linea 10022 è stata data facoltà all'utente di pulire lo schermo. Durante lo sviluppo del programma, quando questo viene fermato e si preme RUN/RESTORE, si possono apportare modifiche al programma stesso senza minimamente influenzare i contenuti dello schermo. Lanciando nuovamente il programma, non dover cancellare gli 8000 byte fa risparmiare tempo.

Linea 10050: questa linea pulisce la normale area di memoria dello schermo, ora utilizzata per contenere i dati di colore di ognuna delle normali 1000 posizioni di carattere.

Collaudo del modulo 3.4.1

Alla prima esecuzione del programma, lo schermo vi apparirà dapprima "sporco". Gradatamente verrà pulito, anche se potrà ancora contenere dei quadrati colorati corrispondenti alla posizione dei caratteri quando lo schermo si trova nel modo normale. Anche questi dovrebbero cominciare a cancellarsi per porre lo schermo sul bianco. Finito il modulo, premete RUN e RESTORE per ritornare al modo consueto.

MODULO 3.4.2

```
11000 REM#********************************
11010 REM DISEGNA SULLO SCHERMO
11020 REM*********************************
11030 X=160:Y=96:MO=1:POKE 1024,(PEEK(10
24)AND240) OR (MO*2)
11040 TT=PEEK(FNPP(X))
11042 GET A$:IF A$<>"" THEN 11050
```

```
11044 POKE FNPP(X),FNPV(X):POKE FNPP(X),
FNPE(X):GOTO 11042
11050 POKE FNPP(X),TT
11060 IF MO<3 THEN X=X-(A$="▌" AND X<319
)+(A$="▐" AND X>0)
11062 IF MO=3 THEN X=X-10*(A$="▌" AND X<
310)+10*(A$="▐" AND X>10)
11070 IF MO<3 THEN Y=Y-(A$="▀" AND Y<191
)+(A$="▄" AND Y>0)
11072 IF MO=3 THEN Y=Y-10*(A$="▀" AND Y<
182)+10*(A$="▄" AND Y>10)
11075 IF A$="*" THEN MO=MO+1:MO=MO+4*(MO
>3):POKE1024,(PEEK(1024)AND240)OR(MO*2)
11080 IF MO=1 THEN POKE FNPP(X),FNPV(X)
11090 IF MO=0 THEN POKE FNPP(X),FNPE(X)
11100 IF A$="1" THEN X1=X:Y1=Y
11110 IF A$="2" THEN X2=X:Y2=Y
11120 IF A$="L" THEN GOSUB 12000
11200 GOTO 11040
11499 GOTO 11499
```

Questo modulo permette di spostare sullo schermo un pixel lampeggiante, attivando o cancellando i singoli pixel.

Commenti

Linea 11030: X e Y sono le coordinate del pixel sullo schermo di estensione 200 * 300. Il cursore a pixel lampeggiante viene posto al centro del video. Nella prima posizione della normale memoria di schermo viene forzato un valore che produce un indicatore colorato del *modo* corrente (nero = 0, rosso = 1, porpora = 2, blu = 3). Gli effetti dei modi verranno spiegati in seguito.

Linea 11040: si calcola lo stato dello schermo nella posizione in cui va fatto lampeggiare il cursore.

Linee 11042-11050: il cursore viene fatto lampeggiare fino alla pressione di un nuovo tasto.

Linee 11060-11072: nel modo 3, premendo i tasti con la freccia di spostamento del cursore, si ottiene il movimento di 10 posizioni del pixel lampeggiante, nella direzione richiesta (all'interno dei limiti dello schermo). Nei modi 0, 1 e 2 si muove solo di uno spazio per volta.

Linea 11075: i tasti funzione senza shift, dall'alto verso il basso, vengono utilizzati per determinare il modo. Cambiando modo, si cambia anche l'indicatore colorato.

Linee 11080-11090: se il modo è zero (nero) il pixel nella posizione del cursore viene cancellato. Nel modo uno (rosso) il pixel viene attivato. I due modi rimanenti permettono il movimento, rapido o lento, del cursore, senza modificare il contenuto dello schermo.

Linee 11100-11120: queste opzioni in ingresso fanno riferimento al modulo successivo.

Collaudo del modulo 3.4.2

Ora dovreste poter spostare sullo schermo il piccolo cursore, disegnando o cancellando.

MODULO 3.4.3

```
12000 REM********************************
12010 REM TRACCIA LINEE
12020 REM********************************
12025 X=X1:Y=Y1
12030 DX=X2-X1+SGN(X2-X1):DY=Y2-Y1+SGN(Y
2-Y1)
12032 IF ABS(DY)>ABS(DX) THEN 12200
12035 SL=ABS(DY/DX)-.5
12040 FORI=1 TO ABS(DX)
12050 IF MODO=1 THEN POKE FNPP(X),FNPV(X
)
12055 IF MODO=0 THEN POKE FNPP(X),FNPE(X
)
12060 IF SL>0 THEN Y=Y+SGN(DY):SL=SL-1:G
OTO12060
12070 SL=SL+ABS(DY/DX)
12100 X=X+SGN(DX):NEXT I
12120 RETURN
12200 SL=ABS(DX/DY)-.5
12210 FOR I=1 TO ABS(DY)
12220 IF MODO=1 THEN POKE FNPP(X),FNPV(X
)
12225 IF MODO=0 THEN POKE FNPP(X),FNPE(X
)
12230 IF SL>0 THEN X=X+SGN(DX):SL=SL-1:G
OTO12230
12240 SL=SL+ABS(DX/DY)
12250 Y=Y+SGN(DY):NEXT I
12300 RETURN
```

Questo modulo dà modo di tracciare linee rette tra due punti definiti dall'utente. Si tratta di un adattamento del metodo noto come algoritmo di Bresenham, e una sua versione viene spesso impiegata nei linguaggi BASIC provvisti di comandi di tracciamento linee.

Commenti

Linea 12025: i valori X1 e Y1 sono stati definiti nel momento in cui l'utente ha battuto "1" — in quel momento sono state loro assegnate le posizioni X e Y del cursore. X2 e Y2 sono stati impostati tramite "2". La linea verrà tracciata partendo da X1 e Y1.

Linea 12030: DX e DY vengono posti uguali alla distanza tra X1 e X2, e Y1 e Y2, più uno. La funzione SGN significa che non ha importanza se la distanza sia positiva o negativa (nel secondo caso si sommerà —1 invece di +1).

Linea 12032: l'algoritmo di tracciamento fa uso della maggiore delle due differenze come base per i suoi calcoli: è dunque più veloce avere due routine separate.

Linea 12035: SL è la pendenza, o il rapporto tra DX e DY meno 0.5.

Linea 12040: il ciclo ha la lunghezza della differenza tra le coordinate X.

Linee 12050-12055: secondo che sia impostato il modo 0 o 1, viene cancellato o disegnato un solo punto della linea. Notate che non succederà niente nei modi 2 o 3.

Linea 12060: a seconda del rapporto tra DX e DY, SL può ora indicare che il punto successivo deve spostarsi in su o in giù lungo l'asse Y. In questo caso si sposta la posizione Y e SL viene ridotto di uno.

Linea 12070: ogni volta che viene impresso un punto, si aggiunge a SL il valore della pendenza.

Linea 12100: la posizione X viene incrementata ad ogni iterazione del ciclo. Una volta ancora la funzione SGN s'incarica di trattare le linee che si muovono verso il basso lungo l'asse.

Linee 12200-12250: esattamente la stessa routine, nei casi in cui DY sia maggiore di DX.

Collaudo del modulo 3.4.3

A questo punto dovreste poter specificare un punto iniziale e uno finale (1 e 2) di una linea, e quindi tracciarla, o cancellare una linea esistente, in conseguenza dell'attivazione del modo 1 o 0.

Questo programma è concepito come non più di un antipasto alle possibilità offerte dal modo bit-map. Un uso completo di questo modo grafico richiede qualche attenta considerazione di quanto desiderate ottenere, e qualche spesso complessa procedura matematica per ottenerlo. Se doveste decidere di proseguire oltre, le tecniche qui fornite e le funzioni utilizzate per localizzare i singoli pixel semplificheranno notevolmente il vostro compito.

Ulteriori sviluppi

1) Perché non aggiungere una funzione che vi permetta il salvataggio su nastro di una pagina grafica? — Vi servirà un nastro piuttosto lungo, ma la routine dovrebbe essere abbastanza semplice.
2) I testi di grafica al calcolatore danno numerosi algoritmi che permettono il tracciamento di circonferenze o archi. Perché non aggiungere alla fine del programma un modulo scritto allo scopo? — Il principale inconveniente sarà la lentezza.

4. Il Commodore 64 come segretario

Presto o tardi, la più parte dei possessori di microcalcolatori comprende che il nuovo amico digitale dà il meglio di sé nell'archiviare informazioni, elaborandole e presentandole in una varietà di modi che, realizzati manualmente, sarebbero estremamente laboriosi. Affronta quindi il compito di scrivere semplici programmi in grado di memorizzare i nomi e gli indirizzi degli amici, o di catalogare la propria collezione di dischi. Può finire con una mezza dozzina di programmi, ciascuno limitato a un singolo uso, e tuttavia ciascuno basato praticamente sugli stessi metodi.
In questo capitolo iniziamo una parte di programmi più sostanziosi, esaminando come si possa scrivere un singolo programma in grado di soddisfare svariate esigenze di archiviazione, senza la costante necessità di riscriverlo ogniqualvolta si presenti l'eventualità di una nuova applicazione.

4.1 Unifile

Il primo programma si chiama Unifile e, nella forma qui presentata, è in grado di archiviare fino a 500 registrazioni[1], oltre a permettere all'utente di ricercare tra queste in modo nominale, di modificare o cancellare delle voci. Al di là delle svariate applicazioni di un simile programma, spero che il semplice fatto di introdurlo in macchina e di capirne i metodi utilizzati vi fornisca numerosi spunti per ulteriori applicazioni.

Unifile: lista delle variabili

IN	Indicatore (*flag*) dell'avvenuta inizializzazione del programma
A%(499,X-1)	Registra la lunghezza delle singole voci di ogni registrazione
A$(499)	Vettore di archiviazione principale

[1] In questo programma, e quando ciò non crei ambiguità nel contesto, si utilizzerà la parola *registrazione* per indicare l'elemento costituito da più *voci*. L'archivio (file) sarà l'insieme delle registrazioni.
In una terminologia più formale le voci andrebbero definite come i *campi* di un *record* (NdT).

B$(X-1)	Contiene i nomi dei tipi delle voci per ogni registrazione
FF	Flag per la determinazione del successo di una ricerca dell'utente
IT	Numero di registrazioni effettuate
PO	Usata nella ricerca binaria per indicare il numero di campionamenti necessari alla ricerca
PP	Puntatore alla posizione iniziale della voce corrente, proveniente da una registrazione, che deve essere stampata
R$	Separatore usato nel salvataggio su nastro dei dati
S1	Puntatore temporaneo di ricerca per il modulo di ricerca da parte dell'utente
SS	Puntatore principale di ricerca nel modulo di ricerca binaria
TI$	Stringa di memorizzazione temporanea usata per costruire una nuova registrazione
TI%(20)	Memorizzazione temporanea della lunghezza delle voci da costruire per una nuova registrazione
X	Contiene il numero di voci specificate per ogni file
Z	Indicatore del numero di funzione di programma da richiamare dal menu principale

MODULO 4.1.1

```
11000 REM#*******************************
11010 REM MENU
11020 REM********************************
11030 POKE 53281,7:PRINT"UNIFILE"
11040 PRINT"FUNZIONI DISPONIBILI:"
11050 PRINT"    1)INTRODUZIONE INFORMA
ZIONI"
11060 PRINT"    2)RICERCA/VISUALIZZAZION
E/MODIFICHE"
11070 PRINT"    3)ARCHIVIAZIONE DATI"
11080 PRINT"    4)COSTRUZIONE NUOVO FILE
"
11090 PRINT"    5)STOP
11100 INPUT "SCEGLIERE LA FUNZIONE DES
IDERATA:";Z:PRINT ""
11110 IF Z>3 OR IN=1 THEN 11140
11120 PRINT "NON ANCORA
INIZIALIZZATA.":FOR I=0 TO 1000:NEXT
11130 GOTO 11000
11140 ON Z GOSUB 13000,17000,18000,12000
,11150:GOTO 11000
11150 PRINT "SISTEMI DI A
RCHIVIAZIONE CHIUSI":END
```

Lo scopo del modulo è di presentare tutte le funzioni rese disponibili dal programma e di permettere all'utente di scegliere tra queste.

In linea di massima, qualsiasi programma complesso, che non inizi con un menu ben formulato delle operazioni che svolge, è un programma di cattiva fattura. Se a questo punto non siete d'accordo con questa affermazione, lo sarete certamente in futuro, quando dovrete tornare a un programma complesso che non è stato usato per alcune settimane e sarete costretti a impiegare molto tempo per scorrere il listato, nel tentativo di ricordare che cosa fa e come.

Commenti

Linea 11030: un tipico uso dei versatili comandi di controllo del cursore del Commodore. La stringa pulisce lo schermo, muove il cursore in giù di una posizione, orizzontalmente fino al centro della linea, attiva la funzione d'inversione (RVS ON) e stampa in verde.

Linee 11110-11130: nessun programma può girare se non sono state costruite le matrici che utilizza. In questo programma la variabile IN viene posta a 1 quando ciò accade. Se IN non è uguale a 1, le sole funzioni del menu disponibili sono l'inizializzazione (costruzione delle matrici) e lo stop.

Linea 11140: per coloro ai quali questo comando risulta nuovo, ON...GOTO e ON...GOSUB sono semplicemente dei modi per accorciare lunghe enumerazioni di enunciati IF...THEN...GOTO (GOSUB). L'istruzione sceglierà nella lista la destinazione designata da Z.

Collaudo del modulo 4.1.1

A questo punto si può solo controllare che il modulo presenti una pagina di menu ben ordinata e che accetti un ingresso. Il solo ingresso che non produrrà un messaggio d'errore è 5 — stop.

MODULO 4.1.2

```
12000 REM#******************************
12010 REM STRUTTURA DEL FILE
12020 REM*******************************
12030 CLR:DIM A$(499):PRINT "
STRUTTURA DEL FILE":IN=1:R$=CHR$(13)
12035 INPUT "CARICAMENTO DAL NASTRO (S
/N):";Q$:IF Q$="S" THEN 11000
12040 INPUT "NUMERO DELLE VOCI IN OGN
I REGISTRAZIONE";X
12043 DIM B$(X-1),A%(499,X-1)
12050 PRINT "";:FOR I=0 TO X-1:PRINT "
NOME DELLA VOCE";STR$(I+1);":";:INPUTQ$
12060 B$(I)=Q$:NEXT I:GOTO 11000
```

Questo modulo compie la fondamentale funzione d'inizializzare le matrici che verranno usate per memorizzare i dati di programma — finché non è stato chiamato, il programma non può essere usato. Una volta introdotti i dati, richiamare il modulo condurrà alla perdita completa dei dati stessi — viene ripulita la memoria, pronta per un nuovo insieme di dati.
L'uso delle principali variabili viene spiegato nella lista delle variabili e nel corso del commento che segue.

Commenti Linea 12030: notate che è necessario ripulire la memoria prima di dimensionare una qualsiasi matrice. Dimenticando di farlo si ha il messaggio d'errore REDIMMED ARRAY (matrice ridimensionata).

Linea 12040: il programma Unifile non impone all'utente il numero di voci che può contenere una registrazione tipo: dipende dall'utente specificarlo. Una volta fatto questo, il programma configura la matrice puntatore A% e il vettore dei nomi delle singole voci B$ di conseguenza.

Linee 12050-12060: specificato il numero di voci in ogni singola registrazione, viene assegnato un nome a ogni voce, per esempio: nome, indirizzo, numero telefonico. Notate che, dal momento che è stata pulita la memoria, questo modulo non può tornare al menu con un RETURN, ma deve essergli assegnata una specifica GOTO.

Collaudo del modulo 4.1.2 Chiamando questo modulo, dovrebbe venirvi chiesto di specificare il numero di voci del file e di assegnare un nome alle singole voci.

MODULO 4.1.3

```
13000 REM#******************************
13010 REM REGISTRAZIONE NUOVE VOCI
13020 REM*******************************
13030 T1$="":PRINT "▮▮▮▮▮▮▮▮▮▮▮VOCI"
13040 PRINT "▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮";IT;"
VOCI PRESENTI"
13050 PRINT "▮▮FUNZIONI DISPONIBILI:"
13060 PRINT "▮▮>▮INTRODUZIONE VOCE SPECI
FICATA"
13070 PRINT "▮>▮BATTERE 'ZZZ' PER TORNAR
E AL MENU▮▮"
13080 FOR I=0 TO X-1:PRINT B$(I);":";:IN
PUT Q$:IF Q$="ZZZ" THEN RETURN
13090 IFLEN(T1$)+LEN(Q$)<=255THEN 13110
13100 PRINT "▮REGISTRAZIONE TROPPO LUNGA
.":FOR J=1 TO 3000:NEXT J:RETURN
13110 T1$=T1$+Q$:TI%(I)=LEN(T1$):NEXT I:
PRINT "▮▮ATTENDERE"
13120 GOSUB14000:GOSUB 15000:GOTO 13000
```

Il compito di questo modulo è accettare in ingresso le voci specificate dall'utente e compilarle in forma di registrazione, pronta per l'archiviazione.

Commenti Linea 13080: usando la variabile X per determinare il numero di iterazioni, il programma richiede all'utente di introdurre ciascuna delle voci nominate.

Linee 13090-13100: ogni singola registrazione può avere lunghezza massima di 255 caratteri — la lunghezza massima di una stringa sul 64. Queste linee controllano che non si superi tale limite.

Linea 13110: la voce introdotta viene aggiunta alla stringa di memorizzazione temporanea T1$ e la lunghezza provvisoria della registrazione viene posta nel vettore TI%. Notate che TI% non era stato dichiarato nel modulo d'inizializzazione. Menzionandolo semplicemente nel corso del programma, lo si dimensiona automaticamente a 10 elementi. Se desiderate avere registrazioni con più di 10 voci, è necessario dichiarare un vettore TI% di maggiori dimensioni nel modulo d'inizializzazione.

Collaudo del modulo 4.1.3

A questo punto, scrivendo dei RETURN temporanei alle linee 14000 e 15000, dovreste poter richiamare questo modulo così che vi sia chiesto di inserire le voci sotto i nomi da voi specificati. Notate come non ci sia ancora alcuna disposizione per inserirle in archivio.

MODULO 4.1.4

```
14000 REM********************************
14010 REM RICERCA BINARIA
14020 REM********************************
14030 IF IT=0 THEN SS=0:RETURN
14040 PO=INT(LOG(IT)/LOG(2)):SS=2↑PO-1
14050 FOR I=PO TO 0 STEP-1
14060 IF A$(SS)<T1$ THEN SS=SS+2↑I
14070 IF A$(SS)>T1$ THEN SS=SS-2↑I
14080 IF SS<0 THEN SS=0
14090 IF SS>IT-1 THEN SS=IT-1
14100 NEXT I:IF A$(SS)<T1$ THEN SS=SS+1
14110 RETURN
```

Tra tutti i moduli di questo programma è quello che probabilmente vi apparirà più astruso. In realtà è molto semplice, ma prima è necessario comprendere i principi basilari che sottostanno al metodo di ricerca noto come ricerca binaria, che riduce in modo stupefacente la quantità di lavoro necessario per trovare il posto giusto di un nuovo elemento in una lista ordinata di dati.

Considerate l'esempio seguente:

Abbiamo costruito un archivio contenente 2000 nomi in ordine alfabetico ed è necessario inserire un nuovo nome, la cui posizione corretta sarà la 1731, cosa tuttavia ancora da determinare. La routine di ricerca inizia dunque esaminando il primo nome del file, stabilisce che il nuovo elemento dovrà essere successivo a questo e si sposta al secondo nome. Alla fine, dopo aver esaminato 1732 nomi, la routine di ricerca ne trova uno che dovrà essere preceduto dal nuovo nome, concludendo così che ha individuato il posto giusto per inserire il nuovo elemento. È questa una procedura diretta e per di più una procedura semplice da implementare, ma confrontatela con quella che segue.

La procedura di ricerca inizia esaminando il nome alla posizione 1024 del file, dato che 1024 è la più alta potenza di 2 che può trovare posto nel numero totale di nomi del file. Il nome 1024 risulta essere

alfabeticamente precedente al nuovo nome, dunque la routine aggiunge 1024/2 ai primitivi 1024 e si porta sul nome numero 1536. Questo precede ancora il nuovo nome, così viene aggiunto 1024/4 a 1536, che fa 1792. Ora accade qualcosa di diverso — il nome numero 1792 è alfabeticamente maggiore del nuovo nome — la soluzione è sottrarre 1024/8, ottenendo 1664. La routine di ricerca continua aggiungendo e sottraendo potenze decrescenti di 2 per ottenere un percorso di ricerca di questo tipo:

1644 (somma 64)
1728 (somma 32)
1760 (sottrai 16)
1744 (sottrai 8)
1736 (sottrai 4)
1732 (sottrai 2)
1730 (somma 1)

Il numero di confronti necessari per trovare il posto esatto nel file è stato ridotto da 1732 a 10. Dovrebbe risultare evidente la potenza del metodo di ricerca binaria.

Linea 14030: non ci sono ancora elementi nel file, dunque la posizione non va calcolata.

Linea 14040: la funzione LOG viene usata per calcolare la massima potenza di 2 contenuta nel numero corrente degli elementi.

Linee 14050-14100: si esegue il salto binario, con due test per controllare che la ricerca non stia superando i limiti del file. Al termine del ciclo viene eseguito un confronto finale e la posizione esatta è determinata e memorizzata nella variabile SS.

Collaudo del modulo 4.1.4

Un collaudo completo di questo modulo dovrà attendere sino all'avvenuta introduzione del successivo, ma un controllo della correttezza sintattica si può avere semplicemente chiamando il modulo d'inserzione, dal quale viene a sua volta chiamato il modulo in esame.

MODULO 4.1.5

```
15000 REM#********************************
15010 REM INSERZIONE
15020 REM*********************************
15030 IF IT=0 THEN GOTO 15060
15040 FOR I=IT TO SS+1 STEP-1:A$(I)=A$(I
-1)
15050 FOR J=0 TO X-1:A%(I,J)=A%(I-1,J):N
EXT J,I
15060 A$(SS)=T1$:FOR I=0 TO X-1:A%(SS,I)
=TI%(I):NEXT:IT=IT+1:RETURN
```

Determinata la posizione esatta, questo modulo sposta tutte le registrazioni da quella posizione in avanti, in su di un posto nell'archivio, contemporaneamente ai puntatori a queste associati, posti in A%.

La nuova registrazione viene inserita nella posizione SS dell'archivio, mentre i puntatori che indicano la lunghezza delle singole voci vengono posti nella corrispondente posizione di A%.

Collaudo dei moduli 4.1.4 e 4.1.5

Dovreste ora poter archiviare delle registrazioni che saranno poste automaticamente in ordine alfabetico. Per controllarlo, dovete fermare il programma e stampare, in modo immediato, i contenuti di A$(0), A$(1) ecc. Dovreste anche controllare che i puntatori, memorizzati alla stessa linea di A%, puntino effettivamente all'ultimo carattere di ogni voce della registrazione.

MODULO 4.1.6

```
18000 REM#*******************************
18010 REM FILE DATI
18020 REM********************************
18030 PRINT "POSIZIONARE IL NASTRO, QU
INDI PREMERE RETURN--"
18040 INPUT "IL MOTORE SI ARRESTA IN MOD
O AUTOMATICO ";Q$:POKE 192,7:POKE 1,39
18050 PRINT "FUNZIONI DISPONIBILI:":PR
INT " 1)SALVATAGGIO DATI"
18055 PRINT " 2)CARICAMENTO DATI"
18060 INPUT "FUNZIONE RICHIESTA:";Q:ON
 Q GOTO 18070,18120:RETURN
18070 POKE 1,7:FOR I=1 TO 2000:NEXT
18080 OPEN 1,1,1,"UNIFILE":PRINT#1,IT,R$
,X
18090 FOR I=0 TO IT-1:PRINT#1,A$(I):FOR
J=0 TO X-1:PRINT#1,A%(I,J):NEXT J,I
18100 FOR I=0 TO X-1:PRINT#1,B$(I):NEXT
18110 CLOSE 1:RETURN
18120 OPEN 1,1,0,"UNIFILE":INPUT#1,IT,X:
IF IN<>1THEN DIM B$(X-1),A%(499,X-1)
18130 FOR I=0 TO IT-1
18132 GET#1,T$:IF T$<>CHR$(13) THEN A$(I
)=A$(I)+T$:GOTO 18132
18134 FOR J=0 TO X-1:INPUT#1,A%(I,J):NEX
T J,I
18140 FOR I=0 TO X-1:INPUT#1,B$(I):NEXT
18150 CLOSE 1:RETURN
```

Ora che potete archiviare dei dati, la prima cosa da fare è memorizzarne alcuni sul nastro, così non dovrete sobbarcarvi al lavoro di reinserire tutti i dati ogni volta che battete nuovi moduli o modificate le linee per correggere degli errori.

Commenti

Linea 18040: posizionato il nastro, ponetelo per prima cosa nel modo RECORD (registrazione) o PLAY (riproduzione) e fatelo scorrere esattamente fino al punto indicato dal contatore del nastro. Raggiunto questo punto, l'azionamento di RETURN, tramite le due POKE, ferma il motore del registratore a cassette.

Linea 18070: viene riacceso il motore del registratore prima di registrare i dati e un'intestazione viene stampata in aggiunta a quella posta automaticamente dal sistema operativo — ciò contribuisce ad essere sicuri di non registrare sul tratto non magnetico del nastro, qualora stiate iniziando dal principio.

Linee 18120-18140: vengono ora richiamati i dati stampati nel file. Notate che, dato che le stringhe del vettore di archivio possono essere più lunghe di 80 caratteri, non è possibile usare un'istruzione INPUT. Ogni carattere delle stringhe di archiviazione viene invece raccolto separatamente usando GET, e ogni riga viene considerata completa quando si raccoglie dal nastro un carattere di ritorno carrello.

Collaudo del modulo 4.1.6

La semplicissima prova di questo modulo si riferisce al fatto di poter introdurre dati nel programma, salvarli su nastro e ricaricarli in seguito.

MODULO 4.1.7

```
17000 REM#********************************
17010 REM RICERCA
17020 REM*********************************
17030 S1=0:FF=0:PRINT "          RIC
ERCA"
17040 PRINT "  FUNZIONI DISPONIBILI:"
17050 PRINT "  > INTRODURRE LA VOCE DA
RICERCARE"
17060 PRINT "  > PREMETTERE 'III' PER RI
CERCA INIZIALE"
17070 PRINT "  > PREMETTERE 'SSS' PER RI
CERCA SPECIALE"
17080 PRINT "  > RETURN  PER LA PRIMA V
OCE DEL FILE"
17090 PRINT "                             
                 ";
17100 T1$="":INPUT"  INTRODURRE LA FUNZ
IONE RICHIESTA:";T1$
17110 IF LEFT$(T1$,3)<>"III" THEN 17140
17120 T1$=RIGHT$(T1$,LEN(T1$)-3):GOSUB 1
4000:S1=SS:IF S1>IT-1 THEN RETURN
17130 GOTO 17240
17140 IF LEFT$(T1$,3)<>"SSS" THEN 17190
17150 FF=0:T1$=RIGHT$(T1$,LEN(T1$)-3):FO
R I=S1 TO IT-1:FOR J=1 TO LEN(A$(I))
17160 IF MID$(A$(I),J,LEN(T1$))=T1$ THEN
 FF=1:S1=I:J=LEN(A$(I)):I=IT-1
17170 NEXT J,I:IF FF=1 THEN T1$="SSS"+T1
$:GOTO 17240
17180 RETURN
17190 IF T1$="" THEN 17240
17200 FF=0:FOR I=S1 TO IT-1:PP=0:FOR J=0
 TO X-1
```

```
17210 IF MID$(A$(I),PP+1,A%(I,J)-PP)=T1$
 THEN FF=1:S1=I:J=X-1:I=IT-1
17220 PP=A%(I,J):NEXT J:NEXT I:IF FF=1 T
HEN 17240
17230 RETURN
17240 IF S1>IT-1 THEN S1=IT-1
17250 IF IT=0 THEN RETURN
17260 IF S1<0 THEN S1=0
17270 PRINT "REGISTRAZIONE ";S1+1;":-
":PP=0
17280 FOR I=0 TO X-1:PRINT "";B$(I);":
";MID$(A$(S1),PP+1,A%(S1,I)-PP)
17290 PP=A%(S1,I):NEXT I:S1=S1+1:PRINT "
"
17300 PRINT "RICERCA FUNZIONI DISPON
IBILI:"
17310 PRINT " >RETURN REGISTRAZIONE
 SEGUENTE"
17320 PRINT " >'AAA' PER CORREGGERE"
17330 PRINT " >'CCC' PER CONTINUARE RI
CERCA"
17340 PRINT " >'#'+NUMERO PER MUOVERE
PUNTATORE"
17350 PRINT " >'ZZZ' PER TORNARE AL ME
NU"
17360 P$="":INPUT "FUNZIONE RICHIESTA:
";P$
17370 IF P$="CCC" THEN 17110
17380 IF P$="" THEN 17240
17390 IF P$="AAA" THEN GOSUB 16000:GOTO
17240
17400 IF P$="ZZZ" THEN RETURN
17410 IF LEFT$(P$,1)="#" THEN S1=S1+VAL(
MID$(P$,2))-1:GOTO 17240
17420 S1=S1-1:GOTO 17240
```

Dopo aver posto i dati nel 64, sarebbe bello pensare di potervi accedere nuovamente. Lo scopo di questo modulo è rendere possibile proprio questo, recuperare le informazioni memorizzate in quella varietà di forme che rendono più utile il sistema d'archiviazione.

Commenti

Linee 17110-17130: se la voce da cercare viene fatta precedere dalle lettere III, si richiama il modulo di ricerca binaria, per trovare una registrazione che inizi con le lettere specificate (oppure la più vicina alla posizione esatta, se non è presente una registrazione corrispondente). Notate che non sarà necessariamente la prima voce dell'archivio a soddisfare la condizione, così, se state usando la funzione di ricerca iniziale per trovare la prima registrazione che inizi, per esempio, con "L", otterrete *una* tale registrazione, e potrete scorrere l'archivio all'indietro per stabilire se si tratti della prima. IIILA

si porterebbe più vicino, mentre IIILAAAA dovrebbe risolvere il problema, a meno che non stiate memorizzando dei nomi particolarmente strani.

Linee 17140-17180: facendo precedere SSS alla voce da ricercare, si avrà il risultato di scandire l'intero archivio in cerca della data combinazione di caratteri — non è necessario che sia una voce completa. SSSLO troverà qualsiasi registrazione contenente LONDRA, LOSANNA o COLONIA. Necessariamente questa ricerca è più lenta di qualsiasi altra.

Linea 17190: se avete premuto RETURN, senza altri ingressi, verrà stampata la prima voce dell'archivio.

Linee 17200-17230: qualsiasi altro ingresso verrà interpretato come una voce completa da ricercare; verranno restituite solo quelle registrazioni che contengono una voce esattamente uguale. Notate, in questa routine, come la matrice dei puntatori A%, che contiene la posizione dell'ultimo carattere di ogni voce di una registrazione, venga utilizzata per estrarre da questa varie voci, anche in assenza di marcatori visibili (nel caso la registrazione venisse stampata direttamente).

Linee 17270-17290: viene stampata sullo schermo la registrazione trovata dal modulo di ricerca.

Linee 17300-17420: visualizzata una registrazione, il programma dà ora facoltà di visualizzare la successiva, di correggere quella trovata, di continuare la ricerca impostata, di spostarsi su un'altra registrazione impostando NN (ove NN è un numero positivo o negativo per muoversi lungo l'archivio), o di ritornare al menu. Se non viene proposto un ingresso riconoscibile, viene stampata di nuovo la stessa registrazione.

Collaudo del modulo 4.1.7

A questo punto dovreste poter visualizzare qualsiasi dato immagazzinato e cercarlo tramite i metodi descritti. Non è ancora possibile correggere le registrazioni.

MODULO 4.1.8

```
16000 REM*********************************
16010 REM MODIFICA REGISTRAZIONI
16020 REM*********************************
16030 S1=S1-1:T1$=""
16040 PP=0:FOR I=0 TO X-1:PRINT "■■■REGI
STRAZIONE ";S1+1;":-■"
16050 PRINT"■";B$(I);":■";MID$(A$(S1),PP
+1,A%(S1,I)-PP)
16060 PRINT "■■■■■■■■■■■■■■■■■■■■■■■MOD
IFICA■"
16070 PRINT "■■FUNZIONI DISPONIBILI:"
16080 PRINT "■ ■>■■RETURN■ VOCE SUCCESSI
VA"
```

```
16090 PRINT " ■>▒INTRODURRE UNA NUOVA VO
CE PER SOSTITUIRE QUELLA VISUALIZZATA"
16100 PRINT " ■>▒'DDD' CANCELLA L'INTERA
 REGISTRAZIONE"
16110 PRINT " ■>▒'ZZZ' NON MODIFICA LA R
EGISTRAZIONE E TORNA A ▒RICERCA■"
16120 Q$="":INPUT "▒▒FUNZIONE RICHIESTA:
";Q$
16130 IF Q$="ZZZ" THEN RETURN
16140 IF Q$="" THEN Q$=MID$(A$(S1),PP+1,
A%(S1,I)-PP)
16150 IF Q$="DDD"THEN GOSUB 16180:RETURN
16160 PP=A%(S1,I):T1$=T1$+Q$:TI%(I)=LEN(
T1$):NEXTI:GOSUB 16180:GOSUB14000
16170 S1=SS:GOSUB 15000:RETURN
16180 FOR J=S1 TO IT-1:A$(J)=A$(J+1):FOR
 K=0 TO X-1:A%(J,K)=A%(J+1,K):NEXT K,J
16190 IT=IT-1:RETURN
```

Lo scopo di questo modulo è di mettervi in grado di apportare modifiche alle voci delle registrazioni già memorizzate, senza doverle ribattere, e di cancellare singole voci o l'intera registrazione, qualora voleste farlo.

Commenti

Linea 16140: il metodo con cui il modulo opera è analogo a quello del modulo d'ingresso principale, con l'eccezione che, se si preme RETURN, la voce introdotta viene definita come voce corrente della visualizzazione.

Linea 16180: la routine sposta verso il basso di una posizione tutte le registrazioni seguenti, cancellando quella corrente.

Collaudo del modulo 4.1.8

Ora dovreste riuscire a correggere voci di una registrazione raggiunta tramite il modulo di ricerca, oppure cancellarla completamente. Se questo modulo funziona come dovrebbe, il programma è pronto all'uso.

Riepilogo

A questo punto avete terminato l'inserimento di un programma complesso e sostanzioso che spero troverete utile in svariate applicazioni. Contemporaneamente avete appreso numerose tecniche che saranno d'aiuto ogniqualvolta deciderete di memorizzare ed elaborare dati non numerici in programmi da voi concepiti.
Se vi siete preoccupati di comprendere, mentre le stavate scrivendo, le funzioni delle singole linee così come quelle complessive dei moduli, dovreste esservi convinti che i programmi complessi e sostanziosi non sono sempre inaccessibili come invece si crede. Tramite un approccio modulare, che divide il programma in una serie di compiti eseguibili, applicazioni come quella presente possono essere sviluppate da parte di chiunque sia pronto a dedicare un po' di tempo allo scopo.

Ulteriori sviluppi

1) Se possedete una stampante, vorrete anche avere la possibilità di stampare su carta singole registrazioni o gruppi di queste. Il modo più semplice sarà l'aggiunta di una funzione alla seconda parte del modulo di ricerca.
2) Una sfida interessante potrebbe consistere nello stabilire se riuscite a dare al programma la capacità di trattare dati sia numerici sia non numerici. Ciò comporterebbe la costruzione di un vettore numerico di 500 elementi, con funzioni per introdurvi valori ed eventualmente qualche comando di ricerca tra le linee, del tipo "trova tutte le registrazioni contenenti valori maggiori di X". Esiste un campo piuttosto vasto di applicazione dove sarebbe vantaggioso poter trattare uno o più elementi di tipo numerico.

4.2 Unifile II

Dopo aver introdotto e collaudato Unifile, l'ultima cosa che potreste voler affrontare è una variazione sul tema. In questo caso sentitevi pienamente liberi di saltare, per ora, questo programma. Vi ritornerete quando sentirete l'esigenza di risolvere almeno alcuni dei problemi che Unifile non è in grado di affrontare. Unifile è adeguato a tutti quei dati con una struttura regolare. Per contro, si hanno numerose applicazioni in cui, semplicemente, non si sa in anticipo quante voci saranno contenute in una particolare registrazione. Potreste, per esempio, voler catalogare i vostri libri. Sarebbe possibile adattare il programma Unifile originale per avere autore e titolo, ma probabilmente, se possedete più di un libro di uno stesso autore, dovreste sprecare tempo e spazio per riportare il nome dell'autore su ogni singolo titolo.
Unifile II è progettato per affrontare simili archiviazioni meno strutturate. Risulta più versatile di Unifile per il fatto che potete continuare ad aggiungere a una registrazione tante voci quante ne volete, entro il limite complessivo di 255 caratteri, e potete definire una più complessa forma di ricerca rivolta a qualsiasi registrazione, contenente fino a 10 obiettivi di ricerca. Questa versatilità comporta, tuttavia, il prezzo di una maggiore difficoltà nell'uso del programma — non sono presenti i pratici messaggi che semplificano l'introduzione delle voci. Inoltre, se volete marcare le voci di una registrazione con un nome, dovete specificarlo e aggiungerlo in forma codificata.
Dal momento che il programma è strutturalmente simile a Unifile, il modo più agevole per introdurlo è cominciare con il caricamento di Unifile stesso. Inserendo Unifile II scoprirete che molte delle linee di programma sono identiche, o quasi, anche se di diversa numerazione. Rinumeratele prima di procedere nelle modifiche, così risparmierete una gran quantità di tempo.

Unifile II: lista delle variabili

B$(49)	Contiene i nomi (opzionali) delle voci
EX	Indicatore temporaneo dell'aggiunta di una voce extra a una registrazione, nel modulo di correzione
FNA(S1)	Funzione che ricava dal valore dell'ultimo carattere di una registrazione il numero di voci ivi presenti

FNB(S1)	Funzione che calcola la posizione dell'ultimo carattere di una voce in una registrazione. Va usata in un ciclo con una variabile che specifichi il numero della voce
NN	Variabile temporanea che indica il numero di voci entro una registrazione da introdurre
SS$	Voce estratta da una registrazione sulla base di FNA e FNB
S2	Puntatore temporaneo usato durante le ricerche
S3	Memorizzazione temporanea del valore di S1 durante ricerche multiple
TI%(49)	Memorizza temporaneamente le posizioni delle voci entro una registrazione da introdurre
TN	Numero di tipo di una voce, se specificato

MODULO 4.2.1

```
11000 REM*********************************
11010 REM MENU
11020 REM*********************************
11030 POKE 53281,7:PRINT"UNIFILE"
11040 PRINT"FUNZIONI DISPONIBILI:"
11050 PRINT"   1)INTRODUZIONE INFORMA
ZIONI"
11060 PRINT"   2)RICERCA/VISUALIZZAZION
E/MODIFICHE"
11070 PRINT"   3)NOMI DI TIPO"
11080 PRINT"   4)ARCHIVIAZIONE DATI"
11090 PRINT"   5)COSTRUZIONE NUOVO FILE
"
11100 PRINT"   6)STOP
11110 INPUT "SCEGLIERE LA FUNZIONE DES
IDERATA:";Z:PRINT ""
11120 IF Z>4 OR IN=1 THEN 11150
11130 PRINT "NON ANCORA
INIZIALIZZATA.":FOR I=0 TO 1000:NEXT
11140 GOTO 11000
11150 ON Z GOSUB 13000,17000,19000,20000
,12000,11160:GOTO 11000
11160 PRINT "SISTEMI DI A
RCHIVIAZIONE CHIUSI":END
```

Un modulo di menu standard.

MODULO 4.2.2

```
12000 REM*********************************
12010 REM INIZIALIZZAZIONE DEL FILE
12020 REM*********************************
12030 CLR:DIM A$(499),B$(49),TI%(49):IN=
1
12040 DEF FNA(S1)=ASC(RIGHT$(A$(S1),1))+
1
```

```
12050 DEF FNB(S1)=ASC(RIGHT$(A$(S1),FNA(
S1)-I+1))
12060 GOTO 11000
```

Inizializza le matrici per tornare subito al menu.

MODULO 4.2.3

```
13000 REM#********************************
13010 REM REGISTRAZIONE NUOVE VOCI
13020 REM*********************************
13030 T1$="":NN=-1:TN=0:PRINT "
VOCI"
13040 PRINT "";IT;"
VOCI PRESENTI"
13050 PRINT "FUNZIONI DISPONIBILI:"
13060 PRINT ">INTRODUZIONE VOCE SPECI
FICATA"
13070 PRINT">'*' PER TERMINARE REGISTR
AZIONE CORRENTE"
13080 PRINT ">'↑NN' PER INTRODURRE UN
NOME DI TIPO"
13090 PRINT ">BATTERE 'ZZZ' PER TORNAR
E AL MENU"
13100 INPUT Q$
13110 IF Q$="ZZZ" THEN RETURN
13120 IF LEFT$(Q$,1)<>"↑" THEN 13150
13130 TN=VAL(MID$(Q$,2)):TN=TN+10
13140 PRINT "";B$(TN-11);":";:GOTO 1310
0
13150 IF TN<>0 THEN Q$=Q$+"↑"+MID$(STR$(
TN),2):TN=0
13160 IFLEN(T1$)+LEN(Q$)+NN+2<=255THEN 1
3180
13170 PRINT "REGISTRAZIONE TROPPO LUNGA
.":FOR J=1 TO 3000:NEXT J:RETURN
13180 IF Q$="*" THEN GOTO 13200
13190 T1$=T1$+Q$:TI%(NN+1)=LEN(T1$):NN=N
N+1:GOTO 13100
13200 FOR I=0 TO NN:T1$=T1$+CHR$(TI%(I))
:NEXT:T1$=T1$+CHR$(NN+1)
13210 PRINT "ATTENDERE":GOSUB14000:GOS
UB 15000:GOTO 13000
```

Questo modulo è equivalente a quello d'ingresso di Unifile, ma è più complesso per due motivi:
1) Devono essere presenti delle funzioni per dire al programma quando è finita una registrazione. Ciò si ottiene tramite un asterisco non seguito da altri dati.
2) Per la mancanza di una struttura regolare, è impossibile fornire dei messaggi regolari per i nomi delle voci da introdurre. Tuttavia nel programma è previsto di marcare le voci. I nomi e i numeri da

assegnare loro vengono definiti in un modulo seguente — in questo modulo è possibile aggiungere il nome a una voce, inserendo prima di tutto il simbolo "↑" seguito dal numero precedentemente assegnato al nome di tipo.

Commenti

Linee 13100-13140: se una registrazione inizia con il simbolo "↑", i caratteri seguenti vengono considerati il numero del nome da associare alla voce da inserire. Il messaggio di input viene ora ripetuto sotto il nome di tipo. Il numero di tipo viene memorizzato al termine della voce e il suo valore viene aumentato di 10 in modo che sia sempre un numero di due cifre (ci sono 50 possibili numeri/nomi di tipo).

Linea 13200: alla stringa di voci costruita vengono ora aggiunti alcuni caratteri il cui valore di codice indica la posizione dell'ultimo carattere di ogni voce. All'estremità finale della stringa viene aggiunto un carattere il cui valore di codice è uguale al numero di voci della registrazione. Notate che, durante il salvataggio su nastro delle registrazioni, questi caratteri vanno tradotti in numeri, dal momento che potrebbero uscire dal campo di quelli che è possibile salvare sotto forma di carattere.

Collaudo del modulo 4.2.3

Non è possibile un controllo completo del modulo, ma si può avere una prova di esecuzione introducendo dei RETURN temporanei alle linee 14000 e 15000. Dovreste quindi riuscire a introdurre delle voci e terminare una registrazione con un asterisco.

MODULO 4.2.4

```
14000 REM#********************************
14010 REM RICERCA BINARIA
14020 REM*********************************
14030 IF IT=0 THEN SS=0:RETURN
14040 PO=INT(LOG(IT)/LOG(2)):SS=2↑PO-1
14050 FOR I=PO TO 0 STEP-1
14060 IF A$(SS)<T1$ THEN SS=SS+2↑I
14070 IF A$(SS)>T1$ THEN SS=SS-2↑I
14080 IF SS<0 THEN SS=0
14090 IF SS>IT-1 THEN SS=IT-1
14100 NEXT I:IF A$(SS)<T1$ THEN SS=SS+1
14110 RETURN
```

Un normale modulo di ricerca binaria come in Unifile.

MODULO 4.2.5

```
15000 REM#********************************
15010 REM INSERZIONE
15020 REM*********************************
15030 IF IT=0 THEN GOTO 15050
15040 FOR I=IT TO SS+1 STEP-1:A$(I)=A$(I
-1):NEXT
15050 A$(SS)=T1$:IT=IT+1:RETURN
```

Un semplice modulo d'inserzione.

Collaudo dei moduli 4.2.4-4.2.5

Ora dovreste poter introdurre delle voci e salvarle nel vettore d'archiviazione principale (A$). Un controllo è possibile solo con i comandi diretti da tastiera.

MODULO 4.2.6

```
19000 REM#*********************************
19010 REM NOMI DI TIPO
19020 REM**********************************
19030 FOR I=0 TO 49 STEP 10
19040 PRINT"█████████████NOMI DI TIPO"
19050 PRINT "██";:FOR J=I TO I+10:PRINT J
+1;")";B$(J):NEXT J
19060 PRINT "██FUNZIONI DISPONIBILI:"
19070 PRINT " █>█'ZZZ' TERMINA"
19080 PRINT " █>█'III' VOCE/CANCELLA"
19090 PRINT " █>█'NNN' PAGINA SUCCESSIVA
"
19100 INPUT "█FUNZIONE RICHIESTA:";Q$:IF
 Q$="ZZZ" THEN RETURN
19110 IF Q$="NNN" THEN NEXT I:RETURN
19120 IF Q$<>"III" THEN GOTO 19000
19130 INPUT "NUMERO DELLA POSIZIONE:";Q
19140 PRINT "█NOME DEL TIPO O █RETURN█ P
ER CANCELLARE:"
19150 Q$="":INPUT Q$:B$(Q-1)=Q$:GOTO1904
0
```

Questo è un modulo nuovo che permette all'utente di definire i tipi delle singole voci. Il modulo non fa altro che visualizzare i contenuti del vettore B$ in gruppi di 11 e dà facoltà all'utente d'inserire un nome in una data posizione della matrice. Una volta introdotto, un nome di tipo si può apporre a una voce, oppure inserire come descritto nel Modulo 4. I nomi si possono ridefinire semplicemente scrivendo un nuovo nome nella posizione occupata dal precedente, oppure si possono cancellare premendo RETURN quando viene richiesto un nuovo nome di tipo.

Collaudo del modulo 4.2.6

Inserito qualche nome di tipo ritornate quindi al modulo d'ingresso principale e premete "↑" seguito dal numero di un tipo definito. Sotto il nome desiderato dovrebbe venir ripetuto il messaggio.

MODULO 4.2.7

```
20000 REM#*********************************
20010 REM FILE DATI
20020 REM**********************************
20030 PRINT "██POSIZIONARE IL NASTRO, QU
INDI PREMERE █RETURN█--"
20040 INPUT "IL MOTORE SI ARRESTA IN MOD
O AUTOMATICO ";Q$:POKE 192,7:POKE 1,39
20050 PRINT "██FUNZIONI DISPONIBILI:":PR
INT "██ 1)SALVATAGGIO DATI"
20060 PRINT " 2)CARICAMENTO DATI"
```

```
20070 INPUT "FUNZIONE RICHIESTA:";Q:ON
 Q GOTO 20080,20160:RETURN
20080 POKE 1,7:FOR I=1 TO 2000:NEXT
20090 OPEN 1,1,1,"UNIFILE":PRINT#1,IT
20100 FORS1=0 TO IT-1:PRINT#1,FNA(S1)
20110 FOR I=1 TO FNA(S1)-1:PRINT#1,FNB(S
1):NEXT I
20120 PRINT#1,LEFT$(A$(S1),LEN(A$(S1))-F
NA(S1)):NEXT S1
20130 FOR I=0 TO 49:IF B$(I)="" THEN B$(
I)=" "
20140 PRINT#1,B$(I):NEXT
20150 CLOSE 1:RETURN
20160 OPEN 1,1,0,"UNIFILE":INPUT#1,IT
20170 FOR S1=0 TO IT-1:INPUT#1,NN
20180 TT$="":FOR I=1TONN-1:INPUT#1,TT:TT
$=TT$+CHR$(TT):NEXTI:TT$=TT$+CHR$(NN-1)
20190 GET#1,T$:IF T$<>CHR$(13) THEN A$(S
1)=A$(S1)+T$:GOTO 20190
20200 A$(S1)=A$(S1)+TT$:NEXT S1
20210 FOR I=0 TO 49:INPUT#1,B$(I):NEXT
20220 CLOSE 1:RETURN
```

Un modulo standard di file-dati.

MODULO 4.2.8

```
21000 REM********************************
21010 REM SUBROUTINE FUNZIONALI
21020 REM********************************
21030 SS$=MID$(A$(S2),PP+1,FNB(S2)-PP):R
ETURN
21040 FF=0:T1$=RIGHT$(T1$,LEN(T1$)-3):FO
R S2=S1 TO IT-1:FOR J=1 TO LEN(A$(S2))
21050 IF MID$(A$(S2),J,LEN(T1$))=T1$ THE
N FF=1:S1=S2:J=LEN(A$(S2)):S2=IT-1
21060 NEXT J,S2:RETURN
21070 FF=0:FOR S2=S1 TO IT-1:PP=0:FOR I=
1 TO FNA(S2)-1:GOSUB 21030
21080 IF SS$=T1$ THEN FF=1:S1=S2:I=FNA(S
2)-1:S2=IT-1:GOTO 21120
21090 IF LEN(SS$)<4 THEN 21110
21100 IF MID$(SS$,LEN(SS$)-2,1)="↑" THEN
 SS$=LEFT$(SS$,LEN(SS$)-3):GOTO 21080
21110 PP=FNB(S2)
21120 NEXT I,S2:RETURN
```

Il modulo consiste di tre routine che vengono vantaggiosamente poste qui, dato che sono richiamate da diverse parti del programma.

Commenti

Linea 21030: può essere richiamata dall'interno di due cicli: un ciclo S2 che specifica la linea di archiviazione principale, e un ciclo

I che determina il numero della voce in una data registrazione. La linea estrae, sotto forma di SS$, la voce PP della registrazione S2.

Linee 21040-21060: equivalenti alla procedura di ricerca speciale di Unifile.

Linee 21070-21120: ricerca diretta della voce. Il ciclo I utilizza FNA per scoprire quante voci ci sono nella registrazione (FNA (S2) = numero di voci in S2 più uno per l'indicatore terminale) e richiama 21030 per estrarre le singole voci. Voci dotate dell'indicatore del tipo vengono confrontate con e senza il suffisso del tipo.

Collaudo del modulo 4.2.8

Impossibile un controllo prima dell'inserzione del modulo seguente.

MODULO 4.2.9

```
17000 REM#**************************
17010 REM RICERCA
17020 REM***************************
17030 S1=0:FF=0:PRINT "■■■■■■■■■■■■■RIC
ERCA"
17040 PRINT "■■FUNZIONI DISPONIBILI:"
17050 PRINT "■■ >■INTRODURRE LA VOCE DA
RICERCARE"
17060 PRINT "■ >■PREMETTERE 'III' PER RI
CERCA INIZIALE"
17070 PRINT "■ >■PREMETTERE 'SSS' PER RI
CERCA SPECIALE"
17080 PRINT "■ >■■RETURN■ PER LA PRIMA V
OCE DEL FILE"
17090 PRINT " ■>■'MMM' PER RICERCA MULTI
PLA"
17100 PRINT "■■■●●●●●●●●●●●●●●●●●●●●●●●●
●●●●●●●●●●●●●●●●]]"
17110 T1$="":INPUT"■INTRODURRE LA FUNZIO
NE RICHIESTA:";T1$
17115 IF LEFT$(T1$,1)<>"↑" THEN 17130
17120 LET TN=VAL(MID$(T1$,2))+10:GOTO 17
110
17130 IF TN<>0 THEN LET T1$=T1$+"↑"+MID$
(STR$(TN),2):TN=0
17140 IF LEFT$(T1$,3)<>"III" THEN 17170
17150 T1$=RIGHT$(T1$,LEN(T1$)-3):GOSUB 1
4000:S1=SS:IF S1>IT-1 THEN RETURN
17160 T1$="III"+T1$:GOTO 17310
17170 IF LEFT$(T1$,3)<>"SSS" THEN17200
17180 GOSUB 21040:IF FF=1 THEN T1$="SSS"
+T1$:GOTO 17310
17190 RETURN
17200 IF LEFT$(T1$,3)<>"MMM" THEN 17270
17210 INPUT "■■NUMERO DI VOCI DA RICERCA
RE:";NN
```

```
17220 FOR K=0 TO NN-1:PRINT "RICERCA VOC
E";K+1;:INPUT M$(K):NEXT K
17230 FOR K=0 TO NN-1:T1$=M$(K):S3=S1:GO
SUB 21070:IF FF=0 THEN RETURN
17240 IF S3<>S1 THEN 17230
17250 NEXT K
17260 LET T1$="MMM":GOTO 17310
17270 IF T1$="" THEN 17310
17280 GOSUB 21070
17290 IF FF=1 THEN 17310
17300 RETURN
17310 IF S1>IT-1 THEN S1=IT-1
17320 IF IT=0 THEN RETURN
17330 IF S1<0 THEN S1=0
17340 PRINT "███REGISTRAZIONE ";S1+1;":-
█":PP=0:S2=S1:FOR I=1 TO FNA(S1)-1
17345 IF I/12=INT(I/12)THEN INPUT "█CONT
INUA";TT$
17350 GOSUB 21030: IF LEN(SS$)<4 THEN 17
380
17360 IF MID$(RIGHT$(SS$,3),1,1)<>"↑" TH
EN 17380
17370 PRINT B$(VAL(RIGHT$(SS$,2))-11);":
";:SS$=LEFT$(SS$,LEN(SS$)-3)
17380 PRINT SS$
17390 PP=FNB(S1):NEXT I:S1=S1+1
17400 PRINT "███RICERCA█ █FUNZIONI DISPO
NIBILI:"
17410 PRINT "█ █>██RETURN█ REGISTAZIONE
SUCCESSIVA"
17420 PRINT "█ >█'AAA' PER CORREGGERE"
17430 PRINT " █>█'CCC' PER CONTINUARE RI
CERCA"
17440 PRINT " █>█'#'+NUMERO PER SPOSTARE
 PUNTATORE"
17450 PRINT " █>█'ZZZ' RITORNA AL MENU"
17460 P$="":INPUT "██FUNZIONE RICHIESTA:
";P$
17470 IF T1$="MMM" AND S1<IT THEN 17230
17480 IF P$="CCC" AND S1<IT THEN 17140
17490 IF P$="" THEN 17310
17500 IF P$="AAA" THEN GOSUB 16000:GOTO
17310
17510 IF P$="ZZZ" THEN RETURN
17520 IF LEFT$(P$,1)="#" THEN S1=S1+VAL(
MID$(P$,2))-1:GOTO 17310
17530 S1=S1-1:GOTO 17310
```

Simile al modulo di ricerca in Unifile, ma comprendente la ricerca multipla e i nomi di tipo assegnati alle voci.

Commenti

Linee 17110-17130: notate il modo in cui questa routine stabilisce se vada ricercata una voce dotata del numero di tipo, e richieda quindi un ingresso sotto quella intestazione, aggiungendo il numero del tipo al termine della voce.

Linee 17140-17160: una ricerca iniziale come in Unifile.

Linee 17170-17190: ricerca speciale che fa uso della routine del modulo precedente.

Linee 17200-17260: la nuova routine di ricerca multipla. Richiede all'utente di specificare il numero di voci da cercare e di introdurre poi le singole voci (non vengono trattati i numeri del tipo). Viene richiamata una procedura di ricerca a 21070-21120. Prima di specificare ogni articolo della ricerca, si prende nota del valore del puntatore di ricerca S1 sotto forma della variabile S3.
Quando la routine a 21070 cede nuovamente il controllo a questa routine, il valore di S3 viene nuovamente confrontato con S1. Se S1 è diverso da S3, è chiaro che le due voci non erano presenti nella stessa registrazione. Non appena trovata *una* delle voci di ricerca specificate, quest'ultima viene riportata alla *prima* voce di ricerca, per assicurarsi che l'intera lista delle voci da ricercare sia confrontata con le voci della registrazione.

Linea 17280: se la ricerca è giunta a questo punto, si assume che l'ingresso sia una voce da ricercare con una procedura normale e viene richiamata la routine a 21070. Notate l'uso del flag (indicatore) FF in tutte queste routine di ricerca per indicare se è stato realmente trovato qualcosa.

Linee 17340-17390: si richiama la routine a 21030 per estrarre le singole voci, iniziando dal principio della registrazione considerata. I nomi di tipo vengono stampati, dove sia presente il simbolo "↑", a due caratteri dalla fine della voce.

Collaudo del modulo 4.2.9

Dovreste ora poter scorrere in avanti e indietro lungo le registrazioni, e compiere ricerche tramite i metodi descritti nei commenti.

MODULO 4.2.10

```
18000 REM##################################
18010 REM COMPATTAZIONE FILE
18020 REM##################################
18030 FOR I=S1 TO IT-1:A$(I)=A$(I+1):NEX
T:IT=IT-1:RETURN
```

Questa procedura di una sola linea compatta l'archivio quando vengono fatte delle cancellazioni.

MODULO 4.2.11

```
16000 REM##################################
16010 REM MODIFICA REGISTRAZIONI
16020 REM##################################
16030 S1=S1-1:T1$="":NN=-1:TN=0
```

```
16040 PP=0:S2=S1:FOR I=1 TO FNA(S1)-1
16050 PRINT "JQREGISTRAZIONE ";S1+1;":-"
16060 GOSUB 21030:IF LEN(SS$)<4 THEN 160
90
16070 IF MID$(SS$,LEN(SS$)-2,1)<>"↑" THE
N 16090
16080 PRINT B$(VAL(RIGHT$(SS$,2))-11);":
";LEFT$(SS$,LEN(SS$)-3):GOTO 16100
16090 PRINT SS$
16100 PRINT"SQQQQQQFUNZIONI DISPONIBILI:
"
16110 PRINT "■ >■■RETURN■ NON MODIFICA L
A VOCE"
16120 PRINT " ■>■TERMINARE CON '*' PER N
UOVA VOCE"
16130 PRINT " ■>■SOSTITUZIONE VOCE VISUA
LIZZATA"
16140 PRINT " ■>■'ZZZ' NON APPORTA MODIF
ICHE"
16150 PRINT " ■>■'DDD' CANCELLA REGISTRA
ZIONE"
16160 PRINT " ■>■'RRR' ELIMINA VOCE DALL
A REGISTRAZIONE"
16170 Q$="":INPUT "■FUNZIONE RICHESTA:";
Q$
16180 IF Q$="ZZZ" THEN RETURN
16190 IF Q$="RRR" THEN GOTO 16300
16200 IF Q$="DDD" THEN GOSUB 18000:RETUR
N
16210 IF LEN(T1$)+LEN(Q$)+NN+2<255 THEN
GOTO 16230
16220 PRINT "■REGISTRAZIONE TROPPO LUNGA
":FOR J=1 TO 3000:NEXT:RETURN
16230 EX=0:IF RIGHT$(Q$,1)="*" THEN EX=1
:Q$=LEFT$(Q$,LEN(Q$)-1)
16240 IF Q$="" THEN T1$=T1$+SS$:TI%(NN+1
)=LEN(T1$):NN=NN+1:GOTO 16300
16250 IF LEFT$(Q$,1)<>"↑" THEN 16280
16260 TN=VAL(MID$(Q$,2))+10:PRINT B$(TN-
11);":";:Q$="":INPUT Q$
16270 Q$=Q$+"↑"+MID$(STR$(TN),2)
16280 T1$=T1$+Q$:TI%(NN+1)=LEN(T1$):NN=N
N+1
16290 IF EX=1 THEN 16050
16300 PP=FNB(S1):NEXT I
16310 FOR I=0 TO NN:T1$=T1$+CHR$(TI%(I))
:NEXT:T1$=T1$+CHR$(NN+1)
16320 PRINT "Q■ATTENDERE":GOSUB 18000:GO
SUB 14000:GOSUB 15000:RETURN
```

Equivalente al modulo di modifica di Unifile.

Commenti

Linea 16320: inserendo una voce che termina con un * si indica che questa va posta nella registrazione prima di quella visualizzata. La variabile EX (EXtra) viene posta a uno per segnalare il fatto. Alla linea 16290 si utilizza questa variabile per assicurarsi di non perdere la voce visualizzata.

Collaudo del modulo 4.2.11

Dovreste ora essere in grado di cancellare registrazioni, modificare o inserire voci. Se tutto funziona correttamente in questo modulo, il programma è pronto all'uso.

Riepilogo

Posto che le loro applicazioni saranno diverse, questo programma conserva tutti i punti di forza dell'Unifile originale, e spero che ne apprezzerete l'utilità.
Inoltre spero che introdurre il programma vi abbia dato una visione chiara dei vantaggi dello stile di programmazione modulare. Basandosi sui moduli di Unifile, la versione originale di questo programma è stata scritta in meno di una mattinata, per il semplice motivo che la limpida struttura messa a disposizione da un programma modulare rende assolutamente chiaro dove vadano apportate le necessarie modifiche.

A patto che non siate pressati da gravissimi problemi di spazio in memoria, risparmierete tempo e fatica di programmazione scrivendo i vostri programmi in ben definite unità funzionali. Questo non solo rende più leggibili i programmi, ma aumenta anche le possibilità che riusciate a richiamare la stessa routine da punti diversi del programma, semplifica la sostituzione di funzioni che pensate di poter migliorare in seguito e, cosa altrettanto importante, rende più agevole l'estrazione di intere parti del programma per successivi utilizzi in altre applicazioni.

Ulteriori sviluppi

1) La routine di ricerca multipla non prevede di specificare i nomi di tipo delle voci. Le linee 17110-17130 danno un chiaro esempio di come sarebbe possibile realizzare ciò.
2) Le basi dei dati di tipo professionale solitamente permettono di ricercare registrazioni contenenti, per esempio, quattro delle otto voci di ricerca presenti. Potreste adattare questo programma alla bisogna.

4.3 NNumer

Dopo aver inserito nel 64 due programmi in grado di fare fronte a numerose necessità nel campo dell'archiviazione di dati non numerici, rivolgiamo ora la nostra attenzione ai problemi dell'archiviazione di numeri. Benché la maggior parte delle applicazioni numeriche vadano specificamente messe in relazione a un particolare problema, NNUMER (forma contratta di Nome e Numero) si avvicina molto ai due programmi Unifile, per il fatto di essere concepito come strumento di uso generale nelle applicazioni dove sia utile memorizzare i nomi di alcune voci, dei valori quantitativi a questi associati e poter sommare le voci in quantità variabile. Nel caso vi rie-

sca difficile immaginarne le applicazioni possibili, potrei forse dire che l'idea da cui è scaturito il programma era quella di un contatore di calorie che permettesse all'utente di memorizzare un dizionario contenente fino a 500 cibi e di calcolare agevolmente il valore calorico dei pasti di un giorno o di una settimana. Questo programma è, senza modifiche, sia in grado di calcolare la lista della spesa sia di aiutare a controllare il peso.
Siccome lo stile del programma è molto simile a quello dei due Unifile, e molte funzioni si assomigliano, i commenti e i suggerimenti per le prove sono stati limitati al minimo indispensabile.

NNumer: lista delle variabili

Variabile	Descrizione
A$(499,1)	Matrice dizionario principale
C(499)	Valori associati alle voci specificate nel dizionario
CT	Variabile temporanea usata per cumulare la somma delle voci della lista corrente
CU	Numero di voci della lista corrente
NN$	Nome generale delle voci da registrare
IN	Flag d'inizializzazione
IT	Numero di voci memorizzate nel dizionario principale
NN	Memorizzazione temporanea del valore associato alla nuova voce da inserire
PO	Usata nel determinare il numero di confronti necessari nel modulo di ricerca binaria
QQ$	Nome generale delle quantità associate alle voci da memorizzare
R$	Separatore usato nel salvataggio del file dei dati
SS	Puntatore per la ricerca binaria
T(49)	Valori associati alle voci memorizzate in T$
T$(49,1)	Memorizzazione della lista corrente
T1$	Memorizzazione temporanea del nome della voce da introdurre
T2$	Memorizzazione temporanea dei valori associati a T1$

MODULO 4.3.1

```
11000 REM*********************************
11010 REM MENU
11020 REM*********************************
11030 POKE 53281,13:PRINT "▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒NNUMER▒"
11040 PRINT "▒▒▒FUNZIONI DISPONIBILI:"
11050 PRINT "▒▒▒  1)VISUALIZZA LISTA COR
RENTE"
11060 PRINT "  2)INSERZIONE NELLA LISTA
CORRENTE"
11070 PRINT "  3)INIZIO NUOVA LISTA"
11080 PRINT "  4)CANCELLAZIONE DA LISTA
CORRENTE"
11090 PRINT "  5)ESPANSIONE DIZIONARIO"
11100 PRINT "  6)VISUALIZZAZIONE DIZIONA
RIO"
```

```
11110 PRINT "  7)ARCHIVIAZIONE DATI"
11120 PRINT "  8)INIZIALIZZAZIONE"
11130 PRINT "  9)TERMINE"
11140 INPUT "FUNZIONE RICHIESTA:";Z:PR
INT"";
11150 IF IN<>0 OR Z=8 OR Z=9 THEN GOTO 1
1180
11160 PRINT "NON A
NCORA INIZIALIZZATA"
11170 FOR I=1 TO 2000:NEXT:GOTO 11000
11180 IF IT>0 OR Z=2 OR Z=3 OR Z=5 OR Z=
7 OR Z=8 OR Z=9 THEN 11210
11190 PRINT "DATI A
SSENTI"
11200 FOR I=1 TO 2000:NEXT:GOTO 11000
11210 ON Z GOSUB 13000,14000,14120,19000
,15000,18000,20000,12000,11230
11220 GOTO 11000
11230 PRINT "
NNUMER"
11240 PRINT "PROGRAMMA TERMIN
ATO":END
```

Un normale modulo di menu.

MODULO 4.3.2

```
12000 REM#******************************
12010 REM INIZIALIZZAZIONE
12020 REM*******************************
12030 CLR:DIM A$(499,1),C(499),T$(49,1),
T(49):CURR=0:IT=0:IN=1:R$=CHR$(13)
12035 INPUT "CARICAMENTO DA NASTRO (S/
N):";Q$:IF Q$="S" THEN 11000
12040 INPUT "NOME DELLE VOCI:";NN$
12050 INPUT "NOME DELLA QUANTITA' ASSOC
IATA:";QQ$:GOTO 11000
```

Inizializzazione delle variabili. Il modulo permette inoltre all'utente di specificare il nome di ogni elemento da memorizzare e il nome generale delle unità di misura, per esempio Cibo/Unità, Prodotto/Tipo di confezione.

MODULO 4.3.3

```
15000 REM#******************************
15010 REM ESTENSIONE DIZIONARIO
15020 REM*******************************
15030 IF IT<500 THEN 15050
15040 PRINT "DIZIONARIO COMPLETATO"
:FOR I=1 TO 2000:NEXT:RETURN
15050 PRINT "NUOVE VOCI DEL DI
ZIONARIO"
15060 PRINT "";NN$;:INPUT ":(NOME O '
```

```
ZZZ' PER TERMINARE):";T1$
15070 IF T1$="ZZZ" THEN RETURN
15080 PRINT "▮";QQ$;:INPUT ":";T2$
15090 PRINT "▮QUANTITA' PER ";T2$;:INPUT
 NN
15100 INPUT "▮▮DATI ESATTI? (S/N) ";Q$:I
F Q$="N" THEN GOTO 15050
15110 GOSUB 16000:GOSUB 17000:IT=IT+1:GO
TO 15050
```

Modulo d'ingresso per il dizionario principale. Specificato il nome generico delle varie voci e delle unità di misura, all'utente si richiede di scrivere il nome della voce, il nome dell'unità di misura e la quantità unitaria fondamentale (per es. calorie, prezzo, volume).

MODULO 4.3.4

```
16000 REM#*********************************
16010 REM RICERCA BINARIA
16020 REM**********************************
16030 IF IT=0 THEN SS=0:RETURN
16040 PO=INT(LOG(IT)/LOG(2)):SS=2↑PO-1
16050 FOR I=PO TO 0 STEP-1
16060 IF A$(SS)<T1$ THEN SS=SS+2↑I
16070 IF A$(SS)>T1$ THEN SS=SS-2↑I
16080 IF SS<0 THEN SS=0
16090 IF SS>IT-1 THEN SS=IT-1
16100 NEXT I
16110 IF A$(SS)<T1$ THEN SS=SS+1
16120 RETURN
```

Un normale modulo di ricerca binaria.

MODULO 4.3.5

```
17000 REM#*********************************
17010 REM INSERZIONE VOCI
17020 REM**********************************
17030 IF IT=0 THEN GOTO 17060
17040 FOR I=IT TO SS+1 STEP-1:A$(I,0)=A$
(I-1,0):A$(I,1)=A$(I-1,1)
17050 C(I)=C(I-1):NEXT
17060 A$(SS,0)=T1$:A$(SS,1)=T2$:C(SS)=NN
:RETURN
```

Un modulo d'inserzione.

MODULO 4.3.6

```
18000 REM#*********************************
18010 REM RICERCA-UTENTE
18020 REM**********************************
18030 SS=0:T1$="*1"
18040 PRINT "▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮RICERCA▮"
18050 PRINT "▮▮NUMERO DELLE VOCI:";SS+1
18060 PRINT "▮▮";NN$;":";A$(SS,0)
```

```
18070 PRINT "";QQ$;":";A$(SS,1)
18080 PRINT "QUANTITA' PER ";A$(SS,1);"
:";C(SS)
18090 PRINT "
"
18100 PRINT "FUNZIONI DISPONIBILI:"
18110 PRINT "  >VOCE DA RICERCARE"
18120 PRINT "  >'*'+NUMERO PER SPOSTARE
IL PUNTATORE"
18130 PRINT "  >'DDD' PER CANCELLARE LA
VOCE"
18140 PRINT "  >'ZZZ' PER TORNARE AL MEN
U"
18150 T1$="":INPUT"FUNZIONE RICHIESTA:
";T1$
18160 IF T1$<>"DDD" THEN 18190
18170 FOR I=SS TO IT-1:A$(I,0)=A$(I+1,0)
:A$(I,1)=A$(I+1,1):C(I)=C(I+1):NEXT
18180 IT=IT-1:GOTO 18040
18190 IF T1$="ZZZ" THEN RETURN
18200 IF LEFT$(T1$,1)<>"*" THEN GOSUB 16
000:GOTO 18220
18210 SS=SS+VAL(MID$(T1$,2))
18220 IF SS>IT-1 THEN SS=IT-1
18230 IF SS<0 THEN SS=0
18240 GOTO 18040
```

Modulo di ricerca principale.

Collaudo dei moduli 4.3.1-4.3.6

A questo punto dovreste riuscire a inserire i dati e a controllare la corretta inserzione nel dizionario principale (ordinato per nome delle voci) usando il modulo di ricerca.

MODULO 4.3.7

```
20000 REM******************************
20010 REM FILE DATI
20020 REM******************************
20030 PRINT "POSIZIONARE IL NASTRO, QU
INDI PREMERE RETURN--"
20040 INPUT "IL MOTORE SI ARRESTA IN MOD
O AUTOMATICO ";Q$:POKE 192,7:POKE 1,39
20050 PRINT "FUNZIONI DISPONIBILI:":PR
INT " 1)SALVATAGGIO DATI"
20060 PRINT " 2)CARICAMENTO DATI"
20070 INPUT "FUNZIONE RICHIESTA:";Q:ON
 Q GOTO 20080,20140:RETURN
20080 POKE 1,7:FOR I=1 TO 2000:NEXT
20090 OPEN 1,1,2,"NNUMER":PRINT#1,NN$R$Q
Q$R$;CU;R$;IT:IF CU=0 THEN 20110
20100 FORI=0 TO CU-1:PRINT#1,T$(I,0)R$T$
(I,1)R$T(I):NEXT
```

```
20110 IF IT=0 THEN 20150
20120 FOR I=0 TO IT-1:PRINT#1,A$(I,0)R$A
$(I,1)R$C(I):NEXT
20130 CLOSE 1:RETURN
20140 OPEN 1,1,0,"NNUMER":INPUT#1,NN$,QQ
$,CU,IT
20150 IF CU=0 THEN 20170
20160 FOR I=0 TO CU-1:INPUT#1,T$(I,0),T$
(I,1),T(I):NEXT
20170 IF IT=0 THEN 20190
20180 FOR I=0 TO IT-1:INPUT#1,A$(I,0),A$
(I,1),C(I):NEXT
20190 CLOSE 1:RETURN
```

Un normale modulo di file dei dati. Procedete al salvataggio dei vostri dati!

MODULO 4.3.8

```
14000 REM******************************
14010 REM ESTENSIONE LISTA CORRENTE
14020 REM******************************
14030 IF CU=50 THEN PRINT "
LISTA CORRENTE COMPLETA":RETURN
14040 PRINT "AGGIUNTE ALLA LISTA
 CORRENTE"
14050 PRINT "";NN$;" ('ZZZ' PER TERMI
NARE):";:INPUT T1$:IF T1$="ZZZ" THEN RET
URN
14060 GOSUB 16000:IF A$(SS,0)=T1$ THEN G
OTO 14080
14070 PRINT "CIBO SCONOSCIUTO, PREGO
CONTROLLARE.":FOR I=1 TO 2000:NEXT:RETUR
N
14080 PRINT "";QQ$;":";A$(SS,1):INPUT "
QUANTITA':";Q
14090 INPUT "DATI ESATTI (S/N):";Q$:I
F Q$="N" THEN 14000
14100 T$(CU,0)=A$(SS,0):T$(CU,1)=STR$(Q)
+" "+A$(SS,1):T(CU)=Q*C(SS):CU=CU+1
14110 GOTO 14000
```

Fino a questo punto avete potuto inserire e manipolare dati nel dizionario principale, ma la questione fondamentale del programma è quanto da questo si può estrarre e porre in una lista temporanea nota come "lista corrente". Questo modulo inizia il processo permettendo di nominare una voce del dizionario e di specificare quante unità di questa vadano aggiunte alla lista corrente. Viene richiamato il modulo di ricerca binaria per controllare l'effettiva presenza nel dizionario della voce introdotta.

MODULO 4.3.9

```
13000 REM#******************************
13010 REM VISUALIZZA LISTA CORRENTE
13020 REM*******************************
13030 IF CU=0 THEN RETURN
13040 PRINT "":CT=0:FOR I=0 TO CU-1:PRI
NT "";NN$;":";T$(I,0)
13050 PRINT "";QQ$;":";T$(I,1):PRINT "
QUANTITA':";T(I)
13060 PRINT "OOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOO"
13070 INPUT "PREMERE RETURN PER VOCE S
UCCESSIVA:";Q$:PRINT ""
13080 PRINT "
                 "
13090 CT=CT+T(I):NEXT I
13100 PRINT "TOTALE:";CT
13110 INPUT "PREMERE RETURN PER TORN
ARE AL MENU";Q$:RETURN
```

Dopo aver introdotto le voci nella lista corrente, questo modulo permette di stamparla sullo schermo e calcola un totale dei valori associati alle voci della lista stessa.

MODULO 4.3.10

```
19000 REM#******************************
19010 REM CANCELLAZIONI LISTA CORRENTE
19020 REM*******************************
19030 IF CU=0 THEN RETURN
19040 FOR I=0 TO CU-1:Q$="":PRINT "";T
$(I,0):PRINT "";T$(I,1)
19050 INPUT"DDD=CANCELLA RETURN=SUCCESS
IVO ZZZ=STOP";Q$:IF Q$="ZZZ"THENRETURN
19060 IF Q$<>"DDD" THEN NEXT I:RETURN
19070 FOR J=1 TO CU-1:T$(J,0)=T$(J+1,0):
T$(J,1)=T$(J+1,1):T(J)=T(J+1):NEXT
19080 CU=CU-1:RETURN
19090 STOP
```

Lo scopo di questo modulo è permettere all'utente di scorrere lungo le registrazioni fino alla lista corrente e di cancellare ad arbitrio. Non si hanno complesse funzioni di ricerca, è solo questione di scorrere ogni voce, una alla volta.

Collaudo dei moduli 4.3.8-4.3.10

Dovreste ora poter costruire una lista corrente, visualizzarla e compiere su questa cancellazioni a piacere.

MODULO 4.3.11

```
14120 REM*******************************
14130 REM INIZIALIZZA LISTA CORRENTE
14140 REM*******************************
14150 FOR I=1 TO 50:T$(I,0)="":T$(I,1)="
":T(I)=0:CU=0:RETURN
```

Dal momento che la lista corrente è intesa solo in funzione temporanea, con frequenti azzeramenti, questo modulo pulisce le matrici della lista corrente e azzera il puntatore alla lista corrente. Se questo modulo funziona correttamente, il programma è pronto all'uso.

Riepilogo

Questo programma è un'ulteriore dimostrazione della potenza della programmazione modulare. Indipendentemente dalla notevole differenza nelle applicazioni, molti dei moduli sono stati estratti, con o senza modifiche, dai due programmi precedenti.
Con il procedere nella vostra carriera di programmatori, imparerete rapidamente che, scritta in unità funzionali debitamente separate l'una dall'altra, una raccolta di metodi operativi è ancora più importante di una raccolta di programmi. Una libreria di programmi vi metterà in posizione favorevole fintantoché non sorga una nuova esigenza applicativa. Una raccolta di metodi, propriamente espressi sotto forma di moduli funzionali, vi permetterà di affrontare quelle nuove applicazioni senza quasi alcuno sforzo. Nuovi metodi potete reperirli ovunque, in riviste e libri come questo, dunque cercate costantemente di seguirli, se appaiono di buona qualità, anche se al momento non riuscite del tutto a coglierne l'importanza. In una o due settimane potreste scoprire che si trattava esattamente di quello che stavate cercando per portare a termine quel programma che vi sta causando tanti grattacapi.

5. Programmi didattici

Un campo in cui i microcomputer stanno già cominciando a lasciare la loro traccia è quello dell'istruzione. Al giorno d'oggi nessuna scuola può dirsi completa senza uno o due calcolatori. Non è tuttavia solo in aula che ha importanza l'apprendimento con l'ausilio del computer — infatti tutti i vantaggi derivanti dalla grande capacità di calcolo si possono avere anche a casa. In questo capitolo ci sono tre programmi che indicano un modello per applicazioni di questo genere, qualsiasi sia la vostra età.

5.1 Multidom

Questo è uno dei miei programmi preferiti. Quando lo scrissi, mi preoccupai solo di far sì che fosse in grado di eseguire il compito per cui era stato progettato. Finché non introdussi una serie di domande e non lo feci provare a varie persone, non mi resi conto che programmi simili rendono l'apprendimento attraente, al pari di un qualsiasi gioco.
Come Unifile, questo è un programma camaleontico, progettato per trasformarsi in relazione alle vostre esigenze. In un istante può essere insegnante di francese, pochi minuti dopo può porre complesse domande sulla storia del XIX secolo.

Multidom: lista delle variabili

AA	Variabile temporanea che memorizza la domanda scelta dall'utente
A$(1,499)	Matrice principale contenente domande e risposte
D(1,9)	Puntatore all'inizio dei gruppi di tipi di elemento nella matrice principale
D$(9)	Vettore contenente i tipi degli elementi
IT	Numero di elementi contenuti nella matrice principale
NA$(1)	Vettore contenente i nomi generici per domande e risposte
P1, P2	Puntatori all'estensione di file da trattare per generare le domande
PP	Puntatore alla risposta sbagliata da selezionare dalla matrice

Q(4)	Usata nella costruzione del testo a risposta multipla
Q1	Posizione della risposta casuale estratta dal file
Q2	Posizione della risposta esatta nella matrice Q di risposte possibili
QT	Numero delle domande poste
QU	Indicatore del tipo di domande richieste nel modulo di generazione delle domande
R$	Separatore nel file dei dati
RR	Risposta esatta
SU	Variabile temporanea usata nel calcolare i gruppi nella matrice D
TY	Numero dei nomi di tipo introdotti

MODULO 5.1.1

```
11000 REM#*******************************
11010 REM MENU
11020 REM*******************************
11030 POKE 53281,13:PRINT "
MULTIDOM"
11040 PRINT "FUNZIONI DISPONIBILI:"
11050 PRINT "  1)INTODUZIONE NUOVE VOC
I"
11060 PRINT "  2)RICERCA/CANCELLAZIONE"
11070 PRINT "  3)INTRODUZIONE NUOVI TIPI
"
11080 PRINT "  4)CREAZIONE DOMANDE"
11090 PRINT "  5)VISUALIZZA O AZZERA PUN
TEGGIO"
11100 PRINT "  6)ARCHIVIAZIONE DATI"
11110 PRINT "  7)INIZIALIZZAZIONE"
11120 PRINT "  8)TERMINE"
11130 INPUT "FUNZIONE RICHIESTA:";Z:PR
INT"";
11140 ON Z GOSUB 13000,16000,12100,17000
,18000,19000,12000,11150:GOTO 11000
11150 PRINT "ORARIO D
I LEZIONE TERMINATO":END
```

Un modulo di menu standard.

MODULO 5.1.2

```
12000 REM#*******************************
12010 REM INIZIALIZZAZIONE
12020 REM*******************************
12030 CLR:DIM A$(1,499),D(1,9),D$(9),Q(4
),NA$(1):R$=CHR$(13)
12040 INPUT "CARICAMENTO DA NASTRO (S/
N):";Q$:IF Q$="S" THEN 11000
12050 PRINT "STRUTTURA DELLA
PROVA"
12060 INPUT "NOME DELLE RISPOSTE:";NA$
(0)
```

```
12070 INPUT "NOME DELLE DOMANDE:";NA$(
1)
12080 INPUT "DATI ESATTI (S/N):";Q$:IF
 Q$="N" THEN 12050
12090 D$(0)="NON CLASSIFICATO":TY=1
12100 PRINT "TIPI"
12110 PRINT "INTRODURRE 'ZZZ' PER TERM
INARE:"
12120 PRINT "TIPI INTRODOTTI FINORA:-"
;:IF TY=1 THEN PRINT "NESSUNO"
12130 IF TY<>1 THEN PRINT "":FOR I=0 TO
 TY-1:PRINT "";I+1;"- ";D$(I):NEXT
12140 INPUT "NUOVO TIPO:";Q$:IF Q$="ZZ
Z" THEN 11000
12150 INPUT "DATI ESATTI (S/N):";Q1$:I
F Q1$="N" THEN 12100
12160 IF TY=10 THEN PRINT "SPAZIO ESAU
RITO":FOR I=1 TO 2000:NEXT:GOTO 11000
12170 D$(TY)=Q$:TY=TY+1:GOTO 12100
```

Questo modulo inizializza le variabili principali e permette di specificare il nome generale delle domande e delle risposte. L'utente può anche specificare fino a nove nomi di ''tipo'', usati in seguito per rendere più difficoltosi i test.

MODULO 5.1.3

```
13000 REM#******************************
13010 REM INTRODUZIONE VOCI NUOVE
13020 REM*******************************
13030 PRINT "VOCI NUOVE
":PRINT "'ZZZ' PER TERMINARE "
13040 IF IT>=500 THEN PRINT "SPAZIO ES
AURITO":FOR I=1 TO 3000:NEXT:RETURN
13050 PRINT "";NA$(0);":";:INPUT T1$:I
F T1$="ZZZ" THEN 13160
13060 PRINT "";NA$(1);":";:INPUT T2$:I
F T2$="ZZZ" THEN RETURN
13070 IF TY=1 THEN T=0: GOTO 13100
13080 PRINT "TIPO:":FOR I=1 TO TY:PRINT
 I;D$(I-1):NEXT I
13090 INPUT "QUALE E':";T:T=T-1
13100 PRINT "NUOVE VOCI"
13110 PRINT "";NA$(0);"";T1$:PRINT "
";NA$(1);"";T2$
13120 PRINT "TIPO";D$(T):INPUT "DA
TI ESATTI (S/N):";Q$
13130 IF Q$<>"S" THEN 13000
13140 D(0,T)=D(0,T)+1:T1$=CHR$(48+T)+T1$
13150 GOSUB 14000:GOSUB 15020:GOTO 13000
13160 SU=0:FOR I=0 TO 9:D(1,I)=SU:SU=SU+
D(0,I):NEXT:RETURN
```

Questo modulo permette d'introdurre domande e risposte sotto le intestazioni specificate dall'utente, permettendo di attribuire un tipo, se i tipi sono stati specificati.

Commenti

Linea 13070: se TY vale 1, non sono stati inseriti nomi di tipo e il tipo viene posto come "Non definito".

Linea 13140: viene incrementato di 1 l'elemento rilevante nella prima colonna della matrice D, registrando il fatto che è stato aumentato il gruppo dei tipi. Il numero del tipo viene apposto all'inizio della risposta, sotto forma di un carattere compreso tra 0 e 9.

Linea 13160: la seconda colonna della matrice D viene disposta in modo da contenere la posizione iniziale di ogni gruppo di tipo.

MODULO 5.1.4

```
14000 REM#*******************************
14010 REM RICERCA BINARIA
14020 REM********************************
14030 IF IT=0 THEN SS=0:RETURN
14040 PO=INT(LOG(IT)/LOG(2)):SS=2↑PO-1
14050 FOR I=PO TO 0 STEP-1
14060 IF A$(0,SS)<T1$ THEN SS=SS+2↑I
14070 IF A$(0,SS)>T1$ THEN SS=SS-2↑I
14080 IF SS<0 THEN SS=0
14090 IF SS>IT-1 THEN SS=IT-1
14100 NEXT I
14110 IF A$(0,SS)<T1$ THEN SS=SS+1
14120 RETURN
```

Un normale modulo di ricerca binaria. Notate che, poiché gli elementi sono ordinati per risposta e le risposte hanno preposto il carattere indicante il gruppo di appartenenza, l'ordinamento effettivo è prima di tutto per tipo — i gruppi di tipo non determinato si troveranno all'inizio del file.

MODULO 5.1.5

```
15000 REM#*******************************
15010 REM INSERZIONE
15020 REM********************************
15030 IF IT=0 THEN 15060
15040 FOR I=IT TO SS+1 STEP-1
15050 FOR J=0 TO 1:A$(J,I)=A$(J,I-1):NEX
T J,I
15060 A$(0,SS)=T1$:A$(1,SS)=T2$:IT=IT+1:
RETURN
```

Un modulo d'inserzione standard.

Collaudo dei moduli 5.1.1-5.1.5

L'introduzione e la memorizzazione di dati nella matrice principale dovrebbero ora essere possibili, con ordinamento secondo numero di tipo.

MODULO 5.1.6

```
19000 REM#*********************************
19010 REM FILE DATI
19020 REM**********************************
19030 PRINT "POSIZIONARE IL NASTRO, QU
INDI PREMERE RETURN--"
19040 INPUT "IL MOTORE SI ARRESTA IN MOD
O AUTOMATICO ";Q$:POKE 192,7:POKE 1,39
19050 PRINT "FUNZIONI DISPONIBILI:":PR
INT " 1)SALVATAGGIO DATI"
19060 PRINT " 2)CARICAMENTO DATI"
19070 INPUT "FUNZIONE RICHIESTA:";Q:ON
 Q GOTO 19080,19150:RETURN
19080 POKE 1,7:FOR I=1 TO 2000:NEXT
19090 OPEN 1,1,2,"MULTIDOM":PRINT#1,IT,R
$,TY
19100 FOR I=0 TO TY-1:PRINT#1,D$(I),R$,D
(0,I),R$,D(1,I):NEXT
19110 FOR I=0 TO IT-1
19120 PRINT#1,A$(0,I),R$,A$(1,I):NEXT
19130 PRINT#1,NA$(0),R$,NA$(1)
19140 CLOSE 1:RETURN
19150 OPEN 1,1,0,"MULTIDOM":INPUT#1,IT,T
Y
19160 FOR I=0 TO TY-1:INPUT#1,D$(I),D(0,
I),D(1,I)
19170 NEXT I
19180 FOR I=0 TO IT-1:INPUT#1,A$(0,I),A$
(1,I):NEXT
19190 INPUT#1,NA$(0),NA$(1)
19200 CLOSE 1:RETURN
```

Un normale modulo di file dei dati.

MODULO 5.1.7

```
16000 REM#*********************************
16010 REM RICERCA UTENTE/CANCELLAZIONE
16020 REM**********************************
16030 SS=0
16040 PRINT "RICERCA"
16050 IF SS>IT-1 THEN SS=IT-1
16060 IF SS<0 THEN SS=0
16070 PRINT "
VOCI=";IT-1
16080 PRINT "FUNZIONI DISPONIBILI:"
16090 PRINT " >RETURN VOCE SUCCESSI
VA"
16100 PRINT " >NUMERO POS/NEG PER MUOV
ERE PUNTATORE"
16110 PRINT " >'DDD' PER CANCELLARE VO
CE"
```

```
16120 PRINT "■ >■'ZZZ' PER TERMINARE"
16130 PRINT "■";:FOR I=1 TO 10
16140 PRINT "■
                    ";:NEXT
16150 PRINT "■TTTTTTTTTTREGISTRAZIONE N.-
";SS+1:PRINT"■■";MID$(A$(0,SS),2)
16160 PRINT "■■";A$(1,SS):PRINT "■■";D$(
VAL(LEFT$(A$(0,SS),1)))
16170 Q1$="":INPUT"■■■■FUNZIONE RICHIEST
A:";Q1$
16180 IF Q1$<>"DDD" THEN 16210
16190 D(0,TE)=D(0,TE)-1:FOR I=SS TO IT-1
:A$(0,I)=A$(0,I+1)
16200 A$(1,I)=A$(1,I+1):NEXT I:IT=IT-1:G
OSUB 13160:GOTO 16040
16210 IF Q1$="ZZZ" THEN RETURN
16220 IF Q1$="" THEN SS=SS+1:GOTO 16040
16230 SS=SS+VAL(Q1$):GOTO 16040
16240 GOTO 16240
```

Un modulo di ricerca-utente di concezione lineare.

MODULO 5.1.8

```
17000 REM********************************
17010 REM DOMANDE CASUALI
17020 REM********************************
17030 QU=0
17040 PRINT "■■■■■■■■■■■■■■■DOMANDE"
17050 PRINT "■■■DESIDERI ESTRARRE LE RIS
POSTE SOLO DA"
17060 INPUT "DOMANDE DELLO STESSO TIPO (
S/N) ";Q$:PRINT "■"
17070 IF Q$="S" THEN QU=1
17080 P1=0:P2=IT:Q1=INT(RND(0)*(IT-1)):Q
2=INT(RND(0)*5):Q(Q2)=Q1
17090 IF QU=0 OR D(0,VAL(LEFT$(A$(0,Q1),
1)))<5 THEN 17110
17100 P1=D(1,VAL(LEFT$(A$(0,Q1),1))):P2=
D(0,VAL(LEFT$(A$(0,Q1),1)))
17110 FOR I=0 TO 4:IF I=Q2 THEN 17150
17120 PP=P1+INT(RND(0)*P2):IF PP=Q(Q2) T
HEN 17120
17130 FORJ=0 TO I:IF PP=Q(J) THEN17120
17140 NEXT J:Q(I)=PP
17150 NEXT I
17160 PRINT "■■■■";NA$(1);":■";A$(1,Q(Q2
))
17170 PRINT "■■■■■■";NA$(0);":■"
17180 FOR I=0 TO 4:PRINT 1+I;")  ";MID$(
A$(0,Q(I)),2):NEXT
```

```
17190 PRINT "QUALE PENSI SIA LA RISPOS
TA ESATTA ?"
17200 INPUT "PREMI IL NUMERO CORRISPON
DENTE ";AA:QT=QT+1
17210 IF AA-1=Q2 THEN 17250
17220 PRINT "SBAGLIATO! LA RISPOSTA ES
ATTA ERA:"
17230 PRINT MID$(A$(0,Q(Q2)),2):GOTO 172
70
17240 POKE53281,0
17250 PRINT "":FOR I=1 TO 11:PRI
NT "                    ESATTO!":NEXT
17260 FOR I=1 TO 15:POKE53281,I:FOR J=1
TO 200:NEXT J,I:RR=RR+1
17270 PRINT "RETURN PER UNA NUOVA DO
MANDA O 'ZZZ' PER TERMINARE "
17280 Q$="":INPUT Q$:IF Q$="ZZZ" THEN RE
TURN
17290 PRINT"";:GOTO 17080
```

È l'unico modulo realmente originale del programma. Il suo scopo è produrre una domanda casuale, visualizzare cinque possibili risposte sullo schermo e ricevere dall'utente un ingresso che specifichi la risposta esatta — o quanto meno cerchi di specificarla.

Commenti

Linee 17050-17070: questa routine determina la difficoltà del test. In un caso le risposte possibili verranno estratte dall'intero archivio e, dato che molte di queste ci appariranno probabilmente inappropriate, il nostro compito di ricerca della risposta esatta sarà reso molto più facile. In alternativa le risposte possono essere estratte solo tra quelle dello stesso tipo, rendendo il compito più arduo, dato che tutte le risposte sembreranno almeno plausibili.

Linea 17080: si estrae dall'archivio una domanda casuale, e una posizione casuale per il suo numero di posizione scelto dalla matrice Q.

Linee 17090-17100: se QU vale 0, oppure se non ci sono cinque risposte nello stesso gruppo, il campo da cui estrarre le possibili risposte alternative è l'intero archivio P1-P2. Se QU vale 1 e ci sono almeno cinque risposte nello stesso gruppo, P1 e P2 vengono riposizionati all'inizio e alla fine del gruppo stesso.

Linee 17110-17150: vengono scelte quattro risposte casuali dal campo determinato da P1 e P2. Si compie un controllo che la risposta selezionata in modo casuale non sia coincidente con la risposta esatta o con un'altra risposta casuale scelta in precedenza.

Linee 17240-17260: l'utente viene premiato da una multicolore serie di cambiamenti dello sfondo dello schermo e viene incrementata la variabile RR.

Collaudo del modulo 5.1.8

Se avete alcuni dati pronti da ricaricare, dovreste ora essere in grado di produrre i vostri test con risposte multiple. Potrete generare la forma più difficoltosa di test solo se avete introdotto dati sufficienti perché il programma possa sempre spaziare su un gruppo di cinque risposte dello stesso tipo.

MODULO 5.1.9

```
18000 REM#******************************
18010 REM PUNTEGGIO
18020 REM*******************************
18030 PRINT "PUNTEGGI
O":IF QT=0 THEN RETURN
18040 PRINT "DOMANDE TOTALI:";QT
18050 PRINT "RISPOSTE ESATTE:";RR
18060 PRINT "PUNTEGGIO:";INT(((RR-QT/5
)/(QT*.8))*100);"%"
18070 INPUT "DESIDERI AZZERARE IL PUNT
EGGIO (S/N) ";Q$
18080 IF Q$="S" THEN QT=0:RR=0
18090 RETURN
```

Questo modulo calcola il punteggio tenendo conto anche del fatto che c'è una probabilità su cinque di dare la risposta giusta per puro caso.

Riepilogo

Si tratta di un programma piuttosto potente; ricordate tuttavia che solo introducendo dati sufficienti per renderlo godibile potrete verificare questa affermazione. Il programma conferma inoltre che, quando possibile, dovendo scrivere un programma complesso, val la pena di impegnarsi per scriverne uno di uso più generale, risparmiando così in futuro parecchio lavoro.

Ulteriori sviluppi

1) Allo stato attuale, il programma controlla che non si stampi due volte la stessa risposta per una sola domanda. Ciò che non fa è invece controllare se due risposte distinte in realtà non siano la stessa. Potreste introdurre un controllo per affrontare questa difficoltà.
2) È interessante la questione della "ricompensa": gli adulti sembrano trovare di per sé gratificante il successo mentre giocano con questo programma. Per i bambini sono tuttavia possibili vari tipi di ricompensa. Che dire dell'aggiunta al programma di un breve gioco, accessibile per tre soli minuti, dopo aver risposto esattamente a un dato numero di domande?

5.2 Parole

Una volta in possesso di un programma di buon funzionamento, scoprite presto che vi suggerisce altri usi. È il caso di Multidom, e il risultato è stato questo programma, utilizzabile come divertente ausilio nell'apprendimento di vocaboli da parte dei bambini nei primissimi approcci con la lettura. La sola concreta differenza tra questo programma e Multidom è che le domande vengono poste sotto forma di disegni, mentre le risposte sono parole che è possibile associare alle figure.

Per quanto riguarda i disegni, non sono altro che il risultato di un altro programma di questo libro, Artista, raccolto dal nastro e caricato nel "dizionario" di questa routine. La capacità di questo programma, nella forma qui presentata, è di 50 disegni, benché sia possibile caricarne dal nastro un altro insieme. I disegni previsti devono occupare solo le dieci linee inferiori dello schermo, dal momento che la parte superiore serve per le domande e le istruzioni destinate all'utente.

Parole: lista delle variabili

A%	Memorizza i dati di caratteri e colori per il disegno
B%	Coordinate degli angoli dei disegni
FNA(SS)	Valore dell'elemento in A% di posizione determinata dalla locazione degli angoli dei disegni
FNB(SS)	Codice di carattere effettivo, ricavato da FNA
WW$	Risposta fornita al modulo delle domande

MODULO 5.2.1

```
11000 REM#********************************
11010 REM MENU
11020 REM*********************************
11030 POKE 53281,15:PRINT "
PARLE"
11040 PRINT "FUNZIONI DISPONIBILI:"
11050 PRINT "  1)INTODUZIONE NUOVE VOC
I"
11060 PRINT "  2)RICERCA/CANCELLAZIONE"
11070 PRINT "  3)CREAZIONE DOMANDE"
11080 PRINT "  4)VISUALIZZA O AZZERA PUN
TEGGIO"
11090 PRINT "  5)ARCHIVIAZIONE DATI"
11100 PRINT "  6)INIZIALIZZAZIONE"
11110 PRINT "  7)TERMINE"
11120 INPUT "FUNZIONE RICHIESTA ";Z:PR
INT"";
11130 ON Z GOSUB 13000,15000,16000,17000
,18000,12000,11140:GOTO 11000
11140 PRINT "
ARRIVEDERCI":END
```

Un normale modulo di menu.

MODULO 5.2.2

```
12000 REM#********************************
12010 REM INIZIALIZZAZIONE
12020 REM*********************************
12030 CLR:DIM A$(49),Q(4),A%(49,255),B%(
49,4):IT=0:R$=CHR$(13)
12040 GOTO 11000
12060 GOTO11000
```

Inizializzazione delle variabili principali.

MODULO 5.2.3

```
13000 REM*******************************
13010 REM INTRODUZIONE VOCI NUOVE
13020 REM*******************************
13030 PRINT "                NUOVE VOCI
"
13040 IF IT>=100 THEN PRINT "  SPAZIO ES
AURITO":FOR I=1 TO 3000:NEXT:RETURN
13050 INPUT "  POSIZIONARE IL NASTRO, QU
INDI PREMERE  RETURN  ";Q$
13060 PRINT " ":OPEN1,1,0,"ARTISTA":FOR
I=0 TO 4:INPUT#1,B%(IT,I):NEXT
13070 FOR I=0 TO B%(IT,4)-1:INPUT#1,C1,C
2
13080 A%(IT,I)=256*C1+C2-32767:NEXT I:CL
OSE1:SS=IT
13090 PRINT" ";:GOSUB 14000
13100 INPUT "  FIGURA ADATTA (S/N)";Q$:I
F Q$="N" THEN RETURN
13110 INPUT "PAROLA DA ASSOCIARE ALLA FI
GURA";W$
13120 INPUT "  DATI ESATTI (S/N):";Q$:IF
 Q$="N"THEN 13090
13130 A$(IT)=W$:IT=IT+1
13140 INPUT "UN' ALTRA FIGURA (S/N) ";Q$
:IF Q$="S" THEN 13000
13150 RETURN
```

Questo modulo raccoglie dal nastro i disegni creati con Artista e permette di aggiungervi la parola esatta.

MODULO 5.2.4

```
14000 REM*******************************
14010 REM DISEGNA
14020 REM*******************************
14030 PP=0:FOR I=B%(SS,1)+1 TO B%(SS,3)-
1
14040 FOR J=B%(SS,0)+1 TO B%(SS,2)-1
14050 PP=PP+1:T1=A%(SS,PP)+32767:POKE 10
24+40*I+J,INT(T1/256)
14060 POKE 55296+40*I+J,T1-256*INT(T1/25
6):NEXT J,I
14070 RETURN
```

Questo modulo fa uso delle due funzioni già definite per estrarre dai valori numerici salvati da Artista i caratteri e i colori esatti e per immetterli nelle memorie di colore e di schermo.

Commenti

Linea 14030: B%(SS,1) e B%(SS,3) registrano le coordinate verticali degli angoli superiore sinistro e inferiore destro del disegno.

Linea 14040: B%(SS,0) e B%(SS,2) registrano le coordinate orizzontali degli angoli anzidetti.

Collaudo dei moduli 5.2.1-5.2.4

A questo punto dovreste essere in grado di caricare i disegni creati da Artista, vederli riprodotti sullo schermo e quindi caricarli nella matrice principale, associati a una data parola.

MODULO 5.2.5

```
18000 REM#*******************************
18010 REM FILE DATI
18020 REM********************************
18030 PRINT "■■POSIZIONARE IL NASTRO, QU
INDI PREMERE ■RETURN■--"
18040 INPUT "IL MOTORE SI ARRESTA IN MOD
O AUTOMATICO ";Q$:POKE 192,7:POKE 1,39
18050 PRINT "■■FUNZIONI DISPONIBILI:":PR
INT "■■ 1)SALVATAGGIO DATI"
18060 PRINT " 2)CARICAMENTO DATI"
18070 INPUT "■■FUNZIONE RICHIESTA:";Q:ON
 Q GOTO 18080,18140:RETURN
18080 POKE 1,7:FOR I=1 TO 2000:NEXT
18090 OPEN 1,1,1,"PAROLE":PRINT#1,IT
18100 FOR I=0 TO IT-1:PRINT#1,A$(I),R$,B
%(I,4)
18110 FOR J=0 TO B%(I,4)-1:PRINT#1,A%(I,
J):NEXT J
18120 FOR J=0 TO 3:PRINT#1,B%(I,J):NEXT
J,I
18130 CLOSE 1:RETURN
18140 OPEN 1,1,0,"PAROLE":INPUT#1,IT
18150 FOR I=0 TO IT-1:INPUT#1,A$(I),B%(I
,4)
18160 FOR J=0 TO B%(I,4)-1:INPUT#1,A%(I,
J):NEXT J
18170 FOR J=0 TO 3:INPUT#1,B%(I,J):NEXT
J,I
18180 CLOSE1:RETURN
```

Un modulo di file dei dati standard.

MODULO 5.2.6

```
15000 REM#*******************************
15010 REM RICERCA UTENTE/CANCELLAZIONE
15020 REM********************************
15030 SS=0:IF IT=0 THEN RETURN
15040 PRINT "■■▶▶▶▶▶▶▶▶▶▶▶▶■■RICERCA"
15050 IF SS>IT-1 THEN SS=IT-1
15060 IF SS<0 THEN SS=0
15070 PRINT "▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶■
VOCI=";IT
15080 PRINT "■■FUNZIONI DISPONIBILI:"
15090 PRINT "■■ >■■RETURN■ VOCE SUCCESSI
VA"
15100 PRINT "■ >■NUMERO POS/NEG PER MUOV
ERE PUNTATORE"
```

```
15110 PRINT "■ >■'DDD' PER CANCELLARE VO
CE"
15120 PRINT "■ >■'ZZZ' PER TERMINARE"
15130 PRINT "■■■■";A$(SS):GOSUB 14000
15140 Q1$="":INPUT"■■■■■■■■■■■■■FUNZIONE
RICHIESTA:",Q1$
15150 IF Q1$<>"DDD" THEN 15190
15160 FOR I=SS TO IT-1:FOR J=0 TO 255:A%
(I,J)=A%(I+1,J):NEXT J,I
15170 FOR I=SS TO IT-1:A$(I)=A$(I+1):FOR
 J=0 TO 4:B%(I,J)=B%(I+1,J):NEXT J,I
15180 IT=IT-1:GOTO 15040
15190 IF Q1$="ZZZ" THEN RETURN
15200 IF Q1$="" THEN SS=SS+1:GOTO 15040
15210 SS=SS+VAL(Q1$):GOTO 15040
15220 GOTO 15220
```

Un modulo di ricerca d'utente di semplice fattura.

MODULO 5.2.7

```
16000 REM#*******************************
16010 REM DOMANDE CASUALI
16020 REM********************************
16030 Q1=INT(RND(0)*IT):Q2=INT(RND(0)*5)
:Q(Q2)=Q1
16040 FOR I=0 TO 4:IF I=Q2 THEN 16090
16050 PP=INT(RND(0)*IT)
16060 IF PP=Q1 THEN 16050
16070 FOR J=0 TO I:IF PP=Q(J) THEN 16050
16080 NEXT J:Q(I)=PP
16090 NEXT I
16100 SS=Q1:PRINT "■":GOSUB 14000
16110 PRINT "■■";:FOR I=0 TO 4:PRINT "■"
;A$(Q(I)):NEXT
16120 PRINT "■■SCRIVI IL NOME ESATTO DEL
LA FIGURA":INPUT WW$:QT=QT+1
16130 IF WW$=A$(Q1)THEN GOTO 16150
16140 PRINT "■■SBAGLIATO! LA RISPOSTA ES
ATTA ERA:■";A$(Q1):GOTO 16170
16150 PRINT "■■ESATTO■ESATTO■ESATTO■ESAT
TO■ESATTO■ESATTO":RR=RR+1
16160 FOR I=1 TO 15:POKE53281,I:FOR J=1
TO 200:NEXT J,I
16170 INPUT "■ANCORA (S/N):";Q$:IF Q$="S
" THEN 16000
16180 RETURN
```

Questo modulo è equivalente al generatore di domande casuali di Multidom, tuttavia è più semplice per il fatto che sceglie le risposte possibili sempre dall'intero archivio delle voci presenti.

MODULO 5.2.8

```
17000 REM#********************************
17010 REM PUNTEGGIO
17020 REM*********************************
17030 IF QT=0 THEN RETURN
17040 PRINT "PUNTEGGI
O"
17050 PRINT "DOMANDE TOTALI:";QT
17060 PRINT "RISPOSTE ESATTE:";RR
17070 PRINT "PUNTEGGIO:";INT(((RR-QT/5
)/(QT*.8))*100);"%"
17080 INPUT "DESIDERI AZZERARE IL PUNT
EGGIO (S/N) ";Q$
17090 IF Q$="S" THEN QT=0:RR=0
17100 RETURN
```

La stessa funzione del modulo per il calcolo del punteggio di Multidom.

Riepilogo

Anche questo è un programma che, se deve avere una qualche utilità, richiede una certa mole di lavoro, dal momento che le figure di cui fa uso, in quantità, richiedono del tempo per la costruzione. Una risposta semplice al problema potrebbe essere quella di unirvi con qualche altro possessore di un 64 per scambiare i nastri dei disegni. Il vostro hobby non deve isolarvi dal resto del mondo!

Ulteriori sviluppi

Il problema della ricompensa riappare in modo ancora più pronunciato — provate ad escogitare modi per cui una risposta esatta possa essere premiata in un modo più stimolante.

5.3 Dattilografo

Non tutti i programmi d'istruzione trattano la manipolazione di dati complessi. Gran parte dei processi di apprendimento si verificano grazie a un costante addestramento delle nostre reazioni, e i computer in questo eccellono; è proprio per questo motivo che astronauti, piloti, ecc. attualmente imparano a svolgere la loro attività di fronte a simulatori computerizzati invece che con il costoso e pericoloso oggetto reale. Questo programma non è ovviamente così ambizioso, ciò nondimeno si dimostra strumento d'apprendimento estremamente efficace.
Tra tutti i programmi che ho scritto, è forse il mio preferito. La sua presenza in questa sede dimostra che un programma non dev'essere necessariamente lungo per essere utile. È breve, "pulito" e adatto a quello che deve fare, cioè ad aiutarmi a migliorare la mia abilità di dattilografo. Di tutte le versioni che ho stilato, quella per il 64 è di gran lunga la migliore, dunque spero che trovi posto nella vostra biblioteca.

Dattilografo: lista delle variabili

C$	Linea di spazi usata per cancellare una linea di testo
CH	Numero di caratteri presenti al momento nelle linee di prova
ESATTO	Numero di caratteri esatti

SOMMA	Numero di caratteri battuti
TI	Variabile di sistema che conta il tempo trascorso in sessantesimi di secondo
TT$	Memorizzazione temporanea del tempo

MODULO 5.3.1

```
11000 REM*********************************
11010 REM VISUALIZZA TASTIERA
11020 REM*********************************
11030 POKE53281,6
11040 PRINT "
"
11050 A$="←1234567890+-£"
11060 PRINT "      ";:FOR I=1 TO 14:PRINT
"  ";MID$(A$,I,1);:NEXT
11070 PRINT "  C  D  "
11080 PRINT "
"
11090 A$="QWERTYUIOP@*↑"
11100 PRINT "       CC";:FOR I=1 TO LEN
(A$):PRINT "  ";MID$(A$,I,1);:NEXT
11110 PRINT "  RR  "
11120 PRINT "
"
11130 A$="ASDFGHJKL:;="
11140 PRINT "      R   S";
11150 FOR I=1 TO LEN(A$):PRINT "  ";M
ID$(A$,I,1);:NEXT
11160 PRINT "  RR  "
11170 A$="ZXCVBNM,./"
11180 PRINT "

11190 PRINT "     R  SH  ";
11200 FOR I=1 TO LEN(A$):PRINT "  ";M
ID$(A$,I,1);:NEXT
11210 PRINT "  SH  ↑  ← "
11220 PRINT "
"
11230 PRINT "
   "
```

Lo scopo di questo modulo è semplicemente far apparire sullo schermo una raffigurazione schematica della tastiera del 64, in modo che si possa guardare lo schermo invece della tastiera, mentre si batte un tasto.

Commenti

Linee 11050-11070: questa sezione, come le seguenti, stampa le parti laterali della tastiera (tasti come RETURN e RESTORE) che non seguono lo schema regolare, stampa quindi una linea di spazi neri inversi a cui sovrappone i nomi dei tasti alfanumerici, nelle posizioni appropriate. Viene poi posta sotto la fila dei tasti una linea nera inversa.

Collaudo del modulo 5.3.1

Questo modulo dovrebbe stampare una copia della tastiera nella metà superiore dello schermo.

MODULO 5.3.2

```
12000 REM#*********************************
12010 REM           INGRESSO
12020 REM**********************************
12030 SOMMA=0:ESATT=0:CH=0:RESTORE:TT$="
000000"
12040 C$="
      "
12050 READ A$:IF A$="STOP" THEN GOTO 120
30
12060 PRINT "■■■■■■■■■■■■■■■■■"
12070 FOR LINEA=1 TO 3:PRINT C$;:NEXT:PR
INT "■■■■"
12080 IF LEN(A$)>39 THEN PRINT "LINEA TR
OPPO LUNGA":STOP
12090 PRINT "■";A$:PRINT "■■■";:FOR I=1
TO LEN(A$)
12100 GET T$:IF T$="" THEN 12100
12110 IF T$="■" OR T$="■" OR T$="■" OR T
$="■" OR T$=CHR$(13) THEN GOTO 12100
12120 IF I=1 THEN TI$=TT$
12130 SOMMA=SOMMA+1:PRINT T$;" ■";
12140 IF T$<>MID$(A$,I,1) THEN PRINT "←■
■";:GOTO 12100
12150 ESATT=ESATT+1:NEXT I:PRINT "■":CH=
CH+LEN(A$):TT$=TI$
12160 PRINT "■■■■";STR$(INT(ESATT/SOMMA*
1000)/10);"%    "
12170 PRINT STR$(INT(SOMMA/(TI/6000))/10
0);" CAR/SEC   "
12180 INPUT "CONTINUA (S/N):";Q$
12190 POKE 780,0:POKE 781,21:POKE 782,0:
SYS 65520:PRINT C$
12200 IF Q$<>"N" THEN 12050
12210 END
```

Questo modulo stampa una linea di testo da copiare, accetta quindi un ingresso tasto per tasto, tenendo conto del tempo, del tasso di successo e indicando gli errori.

Commenti

Linea 12050: il testo da copiare è memorizzato nelle frasi DATA poste alla fine del programma. Queste istruzioni DATA devono essere concluse con una linea contenente semplicemente FINE, come nelle accluse linee d'esempio; ciò conduce il programma nuovamente all'inizio delle READ.

Linee 12060-12090: queste linee utilizzano la stringa di spazi C$ per cancellare l'area dove bisogna stampare il testo, stampano il testo

e spostano la posizione di stampa verso il basso per accettare l'ingresso sulla linea inferiore.

Linea 12110: non vengono accettate come ingresso le frecce di movimento cursore.

Linea 12120: il programma tiene conto del tempo utilizzato per battere il testo, ma il conteggio inizia solo dopo aver battuto la prima lettera di ogni linea e viene sospeso tra le varie linee. Il tempo totale impiegato fino a questo punto viene memorizzato in TT$, valore al quale viene inizializzato TI$ all'inizio di ogni linea (vedi ultimo programma).

Linee 12130-12150: viene stampata l'ultima lettera battuta. Se sbagliata, appare una freccia indicante l'errore e la posizione di stampa viene riportata a quel punto. Vengono registrati il numero totale dei tasti premuti e il numero di quelli giusti.

Linee 12160-12180: dopo aver terminato la linea viene visualizzato il tasso percentuale di successo. La variabile di sistema TI, che contiene lo stesso valore di TI$, espresso però in sessantesimi di secondo, viene utilizzata per calcolare il numero di caratteri battuti al secondo. La formula apparentemente involuta assicura di stampare normalmente due posizioni decimali.

Linea 12190: questa linea mostra un metodo alternativo per determinare la posizione in cui stampare il prossimo carattere. Per fare uso di questo metodo è necessario forzare (tramite le POKE) zero nella locazione 780, la posizione della riga nella 781 e quella della colonna nella 782. Chiamando la routine ROM a 65520, si porta quindi la posizione di stampa in quel punto. Questo metodo si può utilizzare per sostituire stringhe contenenti i caratteri di controllo del cursore. In tal caso viene semplicemente usato per fare sì che il messaggio ANCORA venga sovrastampato con una linea di spazi, se il programma deve continuare.

Collaudo del modulo 5.3.2

Non è possibile collaudare questo modulo finché non siano state inserite alcune DATA che devono essere lette dal programma. Introducete un altro modulo a 13000, consistente nel testo su cui volete esercitarvi. Terminate con una linea DATA contenente FINE, date quindi inizio all'esecuzione del programma. Dovreste trovarvi di fronte la prima linea di testo memorizzata come DATA, ed essere sottoposti al test come descritto nel commento.

Esempio di esercizio

```
13000 REM#******************************
13010 REM DATI DI PROVA
13020 REM*******************************
13030 DATA "ASDF :LKJ ASDF :LKJ ASDF ;LK
J AS"
13040 DATA "ASDF :LKJ ASDF :LKJ ASDF ;LK
J AS"
```

```
13050 DATA "A AD ADD ADDS; ASK LAD ALL F
ALLS"
13060 DATA "A AD ADD ADDS; ASK LAD ALL F
ALLS"
13070 DATA "A AD ADD ADDS; ASK LAD ALL F
ALLS"
13080 DATA STOP
```

Riepilogo

Questo programma può esservi realmente utile solo se lo usate seriamente. Per farne il miglior uso possibile bisogna procurarsi un libro di esercizi di dattilografia e utilizzarlo come base per i dati da introdurre. Per lo sforzo richiesto sarà un efficace strumento per migliorare la vostra padronanza della tastiera.

Ulteriori sviluppi

Una corretta tecnica di dattilografia dipende dall'uso del dito giusto per ogni tasto. Nello stilare un adeguato programma istruttore, dovrebbe essere semplice colorare i tasti della tastiera per indicare il dito da usare. Alla peggio potrebbe essere un buon esercizio nell'uso delle funzioni di colore all'interno di una stringa.

6. Alta micro-finanza

Nonostante i vari aneddoti su bollette del gas da un miliardo di lire, una cosa che i computer fanno in modo superbo è trattare informazioni finanziarie. Ciò non dipende solo dal fatto che sanno memorizzare ed elaborare i dati molto più rapidamente di un essere umano, ma soprattutto dalla loro capacità di presentare i fatti in modo chiaro e comprensibile. In questo capitolo troverete tre programmi di gestione domestica che fanno uso sia delle capacità di calcolo del 64 sia della sua versatile gestione dello schermo, per dissipare, almeno in parte, le nebbie che avvolgono la finanza.

6.1 Banchiere

Il nostro primo programma si chiama appunto Banchiere; è un semplice strumento progettato per tenervi aggiornati circa lo stato delle vostre finanze, prima che la temuta busta della banca scivoli nella cassetta delle lettere. Il programma tratta versamenti e introiti, pagamenti regolari e voci saltuarie, producendo un estratto conto ben impaginato per ciascun mese. Nel corso del programma inizierete ad affrontare alcuni dei problemi connessi alla traduzione di dati numerici sullo schermo in forma comprensibile.

Banchiere: lista delle variabili

A(99,1)	Memorizzazione dell'ammontare dei versamenti/prelievi e del giorno di effettuazione
A$(99,1)	Memorizzazione delle causali e mesi in cui effettuare i versamenti
CD	Indicatore di credito/debito delle operazioni
CR$	Separatore nel file dei dati
IN	Indicatore di avvenuta inizializzazione
M	Numero del mese meno 1
MM	Variabile temporanea utilizzata nel formattare l'ammontare delle operazioni
MM$	Variabile contenente l'ammontare delle operazioni già formattato
MO$	Memorizzazione del nome dei mesi
PA	Numero di operazioni memorizzate

R$	Stringa temporanea contenente i mesi in cui effettuare il versamento
S	Memorizzazione temporanea del giorno del versamento sul mese specificato
SUM	Utilizzata per cumulare le somme, per calcolare il totale dell'estratto conto

MODULO 6.1.1

```
11000 REM#*******************************
11010 REM MENU
11020 REM********************************
11030 POKE53281,7:PRINT "BANCHIERE"
11040 PRINT "FUNZIONI DISPONIBILI:"
11050 PRINT "  1)NUOVI VERSAMENTI"
11060 PRINT "  2)ESAMINA/CANCELLA VERSA
MENTI"
11070 PRINT "  3)STAMPA ESTRATTO CONTO"
11080 PRINT "  4)ARCHIVIAZIONE DATI"
11090 PRINT "  5)INIZIALIZZAZIONE"
11100 PRINT "  6)TERMINA"
11110 INPUT "FUNZIONE RICHIESTA:";Z:P
RINT "";
11120 IF Z=5 OR Z=6 OR IN=1 THEN 11140
11130 PRINT "NON INIZIALIZZA
TO.":FOR I=0 TO2000:NEXT:GOTO11000
11140 IF PA<>0 OR (Z<>2 AND Z<>3) THEN 1
1160
11150 PRINT "DATI NON PRESENTI
.":FOR I=1 TO 2000:NEXT:GOTO 11000
11160 ON Z GOSUB 13000,14000,15000,16000
,12000,11180
11170 GOTO 11000
11180 PRINT "
BANCA"
11190 PRINT "CHIUSA PER AFFARI
"
11200 END
```

Un modulo di menu standard.

MODULO 6.1.2

```
12000 REM#*******************************
12010 REM VARIABILI
12020 REM********************************
12040 CLR:IN=1:DIM A$(99,1),A(99,1):A(0,
1)=999
12050 RESTORE
12060 S$="        "
12070 DIM MO$(11):FOR I=0 TO 11:READ A$:
MO$(I)=A$:NEXT:CR$=CHR$(13)
12080 DATA GENNAIO,FEBBRAIO,MARZO,APRILE
```

```
,MAGGIO,GIUGNO,LUGLIO,AGOSTO
12090 DATA SETTEMBRE,OTTOBRE,NOVEMBRE,DI
CEMBRE
12100 GOTO 11000
```

Inizializza le variabili e pone i nomi dei mesi in MO$.

MODULO 6.1.3

```
13000 REM#*********************************
13010 REM INTRODUZIONE VOCI NUOVE
13020 REM**********************************
13030 PRINT "NUOVE VOC
I"
13040 PRINT "  1)CREDITO":PRINT "  2)DEB
ITO"
13050 INPUT "SPECIFICARE:";CD:CD=CD-1
13060 INPUT "CAUSALE DEL VERSAMENTO:"
;Q$
13070 INPUT "IMPORTO:";Q
13080 INPUT "MESI (P.ES. 01040710):";R$
:PRINT "";
13090 FOR I=1 TO LEN(R$) STEP 2:LET M=VA
L(MID$(R$,I,2))-1
13100 IF M>=0 AND M<=11 THEN GOTO 13130
13110 PRINT:PRINT "VALORE DI MESE NON V
ALIDO                    "
13120 FOR J=1 TO 2000:NEXT:GOTO 13080
13130 PRINT MO$(M);"/";:NEXT:PRINT
13140 INPUT "GIORNO DEL VERSAMENTO:";S
13150 INPUT "DATI ESATTI (S/N):";T$:I
F T$="N" THEN PRINT "";: GOTO 13000
13160 PA=PA+1:FOR J=PA-1 TO 0 STEP-1
13170 IF S<A(J,1) THEN FOR K=0TO1:A$(J+1
,K)=A$(J,K):A(J+1,K)=A(J,K):NEXTK,J
13180 J=J+1:A$(J,1)="000000000000"
13190 FOR I=1 TO LEN(R$) STEP 2:M=VAL(MI
D$(R$,I,2))
13200 A$(J,1)=LEFT$(A$(J,1),M-1)+"1"+RIG
HT$(A$(J,1),12-M):NEXT
13210 A$(J,0)=Q$:A(J,0)=Q:A(J,1)=S
13220 IF CD=1 THEN A(J,0)=A(J,0)*-1
13230 RETURN
```

Lo scopo di questo modulo è permettere l'introduzione della distinta delle operazioni e dei dati associati.

Commenti

Linee 13080-13130: i mesi in cui si compie un pagamento possono variare tra 1, per operazioni saltuarie, e 12, per un ordine fisso. I mesi vengono introdotti sotto forma di una stringa di numeri di due cifre che viene letta e controllata; si stampano quindi sullo schermo

i nomi dei mesi, per controllo. Questo semplice metodo d'ingresso permette una notevole versatilità senza ricorrere a complessi programmi.

Linea 13170: le operazioni vengono memorizzate in una sola matrice, secondo il giorno di effettuazione. L'inserzione si esegue semplicemente scandendo il file dal giorno di valore più elevato.

Linee 13180-13200: queste linee costituiscono un indicatore del mese formato da 12 zeri. Si scandiscono quindi i mesi specificati per porre a 1 le corrispondenti posizioni nell'indicatore.

Linea 13220: introiti e versamenti vengono tutti introdotti come somme positive. Se la variabile CD segnala un debito, l'ammontare viene moltiplicato per meno uno.

MODULO 6.1.4

```
14000 REM********************************
14010 REM ESAMINA/CANCELLA VOCI
14020 REM********************************
14030 FOR I=0 TO PA-1:PRINT "□";
14040 PRINT "■OPERAZIONE:";A$(I,0)
14050 PRINT "■IMPORTO:";A(I,0)
14060 PRINT "■MESI:";:FOR J=1 TO 12
14070 IF MID$(A$(I,1),J,1)="1" THEN PRIN
T MO$(J-1);"/";
14080 NEXT J:PRINT
14090 PRINT "■GIORNO DELL' OPERAZIONE:";
A(I,1)
14100 PRINT "■■■■TASTI FUNZIONALI:■■":P
RINT "■1 - VOCE SEGUENTE"
14110 PRINT "■2 - TERMINA":PRINT "■3 - C
ANCELLA VOCE VISUALIZZATA"
14120 PRINT "■■FUNZIONE RICHIESTA:?■"
14130 GET Q$:IF Q$="" THEN 14130
14140 IF ASC(Q$)<>140 THEN 14170
14150 FOR J=I TO PA-1:FOR K=0 TO 1:A$(J,
K)=A$(J+1,K):A(J,K)=A(J+1,K):NEXT K,J
14160 PA=PA-1:RETURN
14170 IF ASC(Q$)=133 THEN NEXT I
14180 RETURN
```

Un semplice modulo di ricerca-utente che stampa le operazioni e i mesi in cui effettuarle, permettendone la cancellazione tramite tre dei tasti funzione utilizzati per introdurre i comandi.

Collaudo dei moduli 6.1.1-6.1.4

Dovreste a questo punto essere in grado d'introdurre le operazioni e i mesi in cui dovranno essere effettuate, e di scorrere lungo queste, cancellandole a piacere.

MODULO 6.1.5

```
16000 REM#**********************************
16010 REM FILE DATI
16020 REM**********************************
16030 PRINT"POSIZIONARE NASTRO,PREMERE █
RETURN█":INPUT"ARRESTO AUTOMATICO";Q$
16040 POKE192,7:POKE1,39
16050 PRINT "█PREDISPORRE IL REGISTRATOR
E, ":INPUT "PREMERE █RETURN█";Q$
16060 POKE192,7:POKE1,39
16070 PRINT "FUNZIONI DISPONIBILI:":PRIN
T "██1) SALVATAGGIO DATI"
16075 PRINT "2)CARICAMENTO DATI"
16080 INPUT "██FUNZIONE RICHIESTA ";Q:ON
 Q GOTO 16100,16150
16090 RETURN
16100 POKE192,0:FOR I=1 TO 5000:NEXT
16110 OPEN 1,1,2,"BANCA"
16120 PRINT#1,PA
16130 FOR I=0 TO PA-1:PRINT#1,A$(I,0);CR
$;A$(I,1);CR$;A(I,0);CR$;A(I,1):NEXT
16140 CLOSE1:RETURN
16150 OPEN 1,1,0,"BANCA"
16160 INPUT#1,PA
16170 FOR I=0 TO PA-1:INPUT#1,A$(I,0),A$
(I,1),A(I,0),A(I,1):NEXT
16180 CLOSE1
16190 GOTO 11000
```

Un normale modulo di file dei dati.

MODULO 6.1.6

```
17000 REM#**********************************
17010 REM FORMATO
17020 REM**********************************
17030 M$=STR$(MM):HW%=LEN(M$)
17040 FORFF=1TO9-HW%:M$=" "+M$:NEXT:RETU
RN
```

Questo breve modulo è un semplice metodo per ottenere un formato standardizzato per gli importi di denaro da stampare tramite il modulo successivo.

Commenti

Linea 17030: questo modulo viene richiamato da più posizioni del modulo di programma seguente, e opera su valori estratti da variabili diverse. A questo scopo, il valore da formattare deve essere prima memorizzato nella variabile MM. Questa viene ora convertita in una stringa di cui si calcola la lunghezza. Notate che, quando si converte un numero in una stringa tramite STR$, viene automaticamente aggiunto uno spazio iniziale, in modo che il secondo carattere del numero occuperà in realtà la terza posizione della stringa.

Linea 17040: la routine inserisce degli spazi in modo da avere sempre stringhe di lunghezza pari a otto caratteri. Siffatte stringhe si possono stampare in modo da assicurare l'incolonnamento a destra.

MODULO 6.1.7

```
15000 REM*********************************
15005 REM COMPILA ESTRATTO CONTO
15010 REM*********************************
15020 PRINT "ESTRATTO CONTO"
:SOM=0
15030 INPUT "NUMERO DEL MESE DA DOCUMEN
TARE:";Q
15040 IF Q=1 THEN 15080
15050 FOR Q1=1 TO Q-1:FOR I=0 TO PA-1:IF
 MID$(A$(I,1),Q1,1)<>"1" THEN15070
15060 SOM=SOM+A(I,0)
15070 NEXT I,Q1
15080 PRINT "";MO$(Q-1)
15090 PRINT "VOCITO
TALE"
15100 PRINT "BILANCIO C/F
";
15110 IF SOM<0 THEN PRINT ""
15120 LET MM=ABS(SOM):GOSUB 17000:PRINT
M$
15130 FOR I=0 TO PA-1
15140 IF MID$(A$(I,1),Q,1)<>"1" THEN 152
20
15150 LET MM=A(I,1):GOSUB 17000:PRINT "
";M$;
15160 IF A(I,0)<0 THEN PRINT "";
15170 PRINT A$(I,0)
15180 PRINT "";
:MM=ABS(A(I,0)):GOSUB 17000:PRINT M$
15190 SOM=SOM+A(I,0):PRINT "";:IF SOM
<0 THEN PRINT "";
15200 MM=ABS(SOM):GOSUB 17000:PRINT M$
15210 GET A$:IF A$="" THEN 15210
15220 NEXT I:INPUT " RETURN PER CONTIN
UARE:";X
15230 RETURN
```

Questo modulo produce estratti conto mensili di limpida impaginazione — appunto lo scopo del programma.

Commenti

Linee 15030-15070: vengono esaminati tutti gli indicatori del mese delle operazioni memorizzate per stabilire se siano state compiute operazioni sotto l'intestazione di mesi precedenti a quello dell'estratto. Tali operazioni vengono cumulate per avere un bilancio del conto all'inizio del mese specificato.

Linee 15100-15120: si stampa il bilancio, facendo uso del modulo precedente per incolonnarlo. Notate la facilità d'indicare un bilancio negativo, stampando il carattere di controllo per il rosso.

Linee 15130-15220: viene ora esaminato l'archivio delle operazioni che si riferiscono al mese corrente. Ogni volta che si trova un'operazione pertinente, viene stampato il giorno sul lato sinistro dello schermo, utilizzando il modulo di formattazione per standardizzare la stampa. Viene poi stampato il nome dell'operazione, seguito dall'ammontare della stessa, con l'aggiunta del carattere di controllo del rosso se si riferisce a un debito. Al termine l'importo viene aggiunto alla variabile SUM e viene stampato il bilancio corrente accanto all'ammontare dell'operazione, ancora una volta in rosso se il bilancio è negativo. Il versatile controllo di cursore del 64 rende la costruzione di simili tabelle molto agevole — se l'aspetto finale non vi soddisfa, aggiungete semplicemente uno o più spostamenti del cursore fino ad ottenere il risultato voluto.

Linea 15210: le operazioni vengono stampate una alla volta, solo in seguito all'azionamento di uno qualsiasi dei tasti. Si previene in questo modo lo scorrimento verso l'alto oltre l'estremità dello schermo, prima che le scritte possano essere esaminate.

Collaudo dei moduli 6.1.6-6.1.7

Se avete già salvato alcuni dati, dovreste ora essere in grado di ricaricarli nel 64 e di richiedere un estratto conto mensile, con gli importi e le causali dei versamenti incolonnati in modo chiaro. Se l'incolonnamento risulta corretto, il programma è pronto all'uso.

Riepilogo

Questo programma molto lineare solleva alcune domande interessanti circa il grado di raffinatezza necessario a rendere utile un programma. Introdurre i mesi sotto forma di stringa è, per molti aspetti, un'operazione piuttosto rozza a confronto dell'indicare se si deve eseguire il versamento mensilmente, trimestralmente o annualmente, lasciando al programma il compito di inserirlo nei mesi pertinenti. Una tale funzione aggiuntiva sarebbe facilmente realizzabile, ma aumenterebbe la lunghezza del listato e ridurrebbe la versatilità connessa all'indicazione diretta del mese, che permette d'inserire anche mesi irregolari. Nel progettare i vostri programmi, dovete essere sempre consapevoli di questa divergenza tra ciò che è conveniente eseguire in modo automatico e ciò che vale la pena di lasciare all'utente — la risposta può variare secondo l'utente, ma la complessità implicata può risultare costosa in termini di memoria e può effettivamente ridurre l'utilità di un programma.

Ulteriori sviluppi

1) Il modulo di cancellazione è estremamente semplificato, permettendo all'utente di scorrere le registrazioni una per una. Perché non aggiungere una funzione per definire un salto positivo o negativo, usando, per esempio, uno dei programmi precedenti?
2) Un secondo miglioramento potrebbe essere l'aggiunta di un modulo di ricerca binaria, per sostituire l'attuale scansione dalla fine del file durante l'inserzione delle voci.

3) Gli indicatori dei mesi utilizzano 12 byte interi per ogni operazione. Tramite le vostre attuali conoscenze su AND e OR, dovreste poter memorizzare e raccogliere le stesse informazioni con due soli byte (cioè un elemento in una matrice di numeri interi).

6.2 Ragioniere

Questo programma non redigerà i registri in vece vostra, ma li renderà molto più semplici da tenere e li presenterà in un formato ordinato ogniqualvolta lo desideriate, prevedendo, nella stampa di resoconti effettivi, registrazioni singole, voci principali e voci dettagliate.

Ragioniere: lista delle variabili

A$(1,99)	Archivio principale dei nomi delle operazioni
A(1,99)	Archivio principale degli importi delle operazioni
C$	Linea di spazi utilizzata nella cancellazione di testi
C(1)	Matrice contenente il numero delle registrazioni nella lista crediti/debiti dei rendiconti
CD	Indicatore di credito/debito di una operazione
CR$	Separatore nel file dei dati
GR	Utilizzata per registrare il numero di operazioni sotto una singola voce
HH$	Memorizzazione temporanea del nome di una voce principale nel modulo di ricerca - utente
IN	Indicatore di avvenuta inizializzazione
M$	Stringa contenente l'importo formattato
MM	Variabile temporanea utilizzata nel formattare gli importi
PL	Posizione nel file per l'inserzione di una nuova voce dettagliata
SS	Variabile temporanea utilizzata per cumulare le registrazioni sotto una stessa voce principale
TT	Utilizzata per cumulare le registrazioni nei rendiconti

MODULO 6.2.1

```
11000 REM#*********************************
11010 REM MENU
11020 REM**********************************
11030 POKE53281,7:PRINT "
RAGIONIERE"
11040 PRINT "FUNZIONI DISPONIBILI:"
11050 PRINT "  1)NUOVE INTESTAZIONI"
11060 PRINT "  2)ESAMINA/CANCELLA VOCI"
11070 PRINT "  3)STAMPA ESTRATTO CONTO"
11080 PRINT "  4)ARCHIVIAZIONE DATI"
11090 PRINT "  5)INIZIALIZZAZIONE"
11100 PRINT "  6)TERMINA"
11110 INPUT "FUNZIONE RICHIESTA:";Z:P
RINT "";
11120 IF Z<1 OR Z>4 OR IN=1 THEN 11140
11130 PRINT "NON INIZIALIZZA
TO.":FOR I=0 TO2000:NEXT:GOTO11000
```

```
11140 IF Z<4 AND Z>0 THEN GOSUB 13000:PR
INT "";
11150 ON Z GOSUB 14000,17000,19000,20000
,12000,11170:Z=0:GOTO 11000
11170 PRINT "PROGRAM
MA TERMINATO":END
```

Un modulo di menu standard.

MODULO 6.2.2

```
12000 REM#********************************
12010 REM VARIABILI
12020 REM*********************************
12040 CLR:DIM A$(1,99),A(1,99):CR$=CHR$(
13)
12050 C$="
          "
12060 IN=1:GOTO 11000
```

Modulo d'inizializzazione.

MODULO 6.2.3

```
13000 REM#********************************
13010 REM CREDITO / DEBITO
13020 REM*********************************
13030 PRINT "1) CREDITO
2) DEBITO"
13040 INPUT "SPECIFICARE:";CD:
CD=CD-1:RETURN
```

Prima di una qualsiasi registrazione si domanda all'utente di specificare se si tratti di un credito o di un debito.

MODULO 6.2.4

```
14000 REM#********************************
14010 REM INTESTAZIONI
14020 REM*********************************
14030 PRINT "NUOVE VOCI:";:IF
 CD=0 THEN PRINT "A CREDITO"
14040 IF CD=1 THEN PRINT "A DEBITO"
14050 PRINT "LA VOCE PUO' ESSERE:":PR
INT " 1) UNA VOCE SINGOLA"
14060 PRINT " 2)UNA VOCE PRINCIPALE":PR
INT " 3)UNA VOCE DETTAGLIATA"
14070 PRINT " PREMERE "0" PER TERMINARE
"
14080 INPUT "SPECIFICARE:";TT
14090 ON TT GOTO 15000,15000,16000
14100 RETURN
```

Quando viene inserito un elemento, si domanda all'utente di specificare se si tratti di una voce principale, di una voce dettagliata o di una registrazione singola. Quando vengono stampati i rendicon-

ti, le voci principali non sono accompagnate da alcun totale, le voci dettagliate sono poste sotto la rispettiva voce principale e viene stampato un sub-totale per il gruppo, mentre le registrazioni singole si trovano da sole, insieme con una somma.

MODULO 6.2.5

```
15000 REM#*******************************
15010 REM VOCI SINGOLE O PRINCIPALI
15020 REM********************************
15030 Q=0:INPUT "NOME DELLA VOCE:";Q$
15040 IF TT<>2 THEN INPUT "IMPORTO:";Q
15050 INPUT "DATI ESATTI (S/N):";R$:IF
 R$="N" THEN 14000
15060 Q$="%"+Q$:IF TT=2 THEN Q$="*"+MID$
(Q$,2)
15070 A$(CD,C(CD))=Q$:A(CD,C(CD))=Q:C(CD
)=C(CD)+1:GOTO 14000
```

Questo modulo riceve l'ingresso di voci principali o singole registrazioni.

Commenti

Linea 15040: si richiede un importo solo se l'elemento non è una intestazione principale.

Linea 15060: si appone un indicatore all'inizio dell'elemento: % per una registrazione singola, * per una voce principale. Questi non verranno stampati, ma saranno utilizzati dal programma per identificare i vari tipi.

Linea 15070: notate l'uso della variabile CD per specificare su quale colonna della matrice principale verranno memorizzati il nome e l'importo.

MODULO 6.2.6

```
16000 REM#*******************************
16010 REM VOCI DETTAGLIATE
16020 REM********************************
16030 INPUT "NOME DELLA VOCE PRINCIPAL
E:";Q$:Q$="*"+Q$
16040 FOR I=0 TO C(CD)-1:IF A$(CD,I)=Q$
THEN 16060
16050 NEXT:PRINT "VOCE NON ESISTENTE":FO
R I=1 TO 2000:NEXT:GOTO 16000
16060 PL=I+1:INPUT "NOME DELLA VOCE DE
TTAGLIATA:";Q$
16070 INPUT "IMPORTO PER LA VOCE INTROD
OTTA:";Q
16075 PRINT "2)CARICAMENTO DATI"
16080 INPUT "DATI ESATTI (S/N)";R$:IF
R$="N" THEN GOTO 14000
16090 Q$="$"+Q$
16100 FOR I=C(CD)+1 TO PL+1 STEP-1:A$(CD
,I)=A$(CD,I-1):A(CD,I)=A(CD,I-1):NEXT
```

```
16110 A$(CD,PL)=Q$:A(CD,PL)=Q:C(CD)=C(CD
)+1:GOTO 14000
```

Questo modulo accetta l'ingresso delle voci dettagliate.

Commenti

Linee 16030-16050: si domanda il nome della voce principale pertinente, e lo si confronta con i nomi presenti nel file. Se non viene trovato, si genera un messaggio d'errore. Notate come si utilizza un ciclo per condurre la ricerca: il programma esce dal ciclo quando viene trovato l'elemento, e si usa il valore della variabile corrente del ciclo, I, per determinare il punto di inserimento della voce dettagliata. La conclusione del ciclo significa che non è stata trovata la voce principale.

Linee 16090-16110: la voce dettagliata viene marcata con un simbolo $ e viene aggiunta al file principale immediatamente dopo l'intestazione della voce principale.

Collaudo dei moduli 6.2.1-6.2.6

A questo punto dovreste essere in grado d'introdurre nel rendiconto registrazioni a vostro credito o debito e vederle correttamente inserite nella giusta colonna dell'archivio principale (colonna credito = 0, colonna debito = 1). Tutto ciò si può provare solo tramite comandi diretti da tastiera.

MODULO 6.2.7

```
20000 REM********************************
20010 REM FILE DATI
20020 REM********************************
20030 PRINT"POSIZIONARE NASTRO,PREMERE ▒
RETURN■":INPUT"ARRESTO AUTOMATICO";Q$
20040 POKE192,7:POKE1,39
20050 PRINT "FUNZIONI DISPONIBILI:":PRIN
T "▒▒1) SALVATAGGIO DATI"
20055 PRINT "2) CARICAMENTO DATI"
20060 INPUT "▒▒FUNZIONE RICHIESTA ";Q:ON
 Q GOTO 20080,20140
20070 RETURN
20080 FOR I=0 TO 1:IF A$(I,0)="" THEN A$
(I,0)=" "
20090 POKE1,7:FOR I=1 TO 2000:NEXT
20100 OPEN 1,1,2,"CONTI"
20110 FOR I=0 TO 1:PRINT#1,C(I):IF C(I)=
0 THEN 20130
20120 FOR J=0 TO C(I)-1:PRINT#1,A$(I,J);
CR$;A(I,J):NEXT J
20130 NEXTI:CLOSE1:RETURN
20140 OPEN 1,1,0,"CONTI"
20150 FOR I=0 TO 1:INPUT#1,C(I):IF C(I)=
0 THEN 20170
20160 FOR J=0 TO C(I)-1:INPUT#1,A$(I,J),
A(I,J):NEXT J
20170 NEXT I:CLOSE 1:RETURN
```

Un normale modulo di file dei dati.

MODULO 6.2.8

```
21000 REM*********************************
21010 REM FORMATO
21020 REM*********************************
21030 MM=MM+100000000:M$=MID$(STR$(MM),3
,8)
21040 FOR P=1 TO 8
21050 IF MID$(M$,P,1)="0" THEN M$=LEFT$(
M$,P-1)+" "+RIGHT$(M$,8-P):NEXT
21060 RETURN
```

Questo modulo esegue la stessa funzione del modulo di formattazione del programma precedente.

MODULO 6.2.9

```
17000 REM*********************************
17010 REM MODIFICHE E CANCELLAZIONI
17020 REM*********************************
17030 FOR I=0 TO C(CD)-1
17040 PRINT "■■■■■■■■■CANCELLAZIONI O M
ODIFICHE■"
17050 IF LEFT$(A$(CD,I),1)<>"$" THEN PRI
NT "■■",MID$(A$(CD,I),2);
17060 IF LEFT$(A$(CD,I),1)="*" THEN LET
HH$=MID$(A$(CD,I),2):PRINT
17070 IF LEFT$(A$(CD,I),1)="$" THEN PRIN
T "■■";HH$:PRINT"■";MID$(A$(CD,I),2);
17080 IF A(CD,I)=0 THEN 17100
17090 PRINT"■■■■■■■■■■■■■■■";:MM=A(CD,I):
GOSUB 21000:PRINTM$
17100 PRINT "■■■■■■■■■■TASTI FUNZIONALI:
"
17110 PRINT "■■■  F1 - VOCE SUCCESSIVA"
17120 PRINT "■  F3 - MODIFICA IMPORTO"
17130 PRINT "■  F5 - TORNA AL MENU"
17140 PRINT "■  F8 - CANCELLA REGISTRAZI
ONE"
17150 PRINT "■■■FUNZIONE RICHIESTA:?"
17160 GET Q$:IF Q$="" THEN 17160
17170 IF Q$=CHR$(140) THEN GOSUB 18000:R
ETURN
17180 IF Q$=CHR$(135) THEN RETURN
17190 IF Q$<>CHR$(134) OR LEFT$(A$(CD,I)
,1)="*" THEN GOTO 17240
17200 INPUT "■■■IMPORTO DA AGGIUNGERE:";
Q
17210 INPUT "■■DATI ESATTI (S/N):";R$
17220 IF R$<>"N" THEN A(CD,I)=A(CD,I)+Q:
GOTO 17040
17230 PRINT "TTTT";:FOR L=1 TO 3:PRINT C
```

```
$;:NEXT:PRINT "TTTT";:GOTO 17200
17240 NEXT I:RETURN
```

Questo modulo permette di scorrere lungo le registrazioni sulla colonna desiderata del rendiconto e di modificare o cancellare delle registrazioni.

Commenti

Linee 17050-17090: le condizioni presenti in queste istruzioni trattano la formattazione dei vari tipi di elementi. Se si tratta di un elemento diverso da una voce dettagliata, il nome viene stampato e, se si tratta inoltre di voce principale, memorizzato in HH$. Se è invece una voce dettagliata, sopra questa viene stampato il nome della voce principale (precedentemente memorizzato) in modo da indicare il gruppo di appartenenza. Infine, se l'elemento è una registrazione singola oppure una voce dettagliata, si utilizza il modulo precedente per la formattazione prima di stampare l'importo associato.

Linee 17190-17230: premendo il tasto f3 per una voce singola o una registrazione dettagliata, l'utente ha facoltà di modificare l'importo associato all'elemento, introducendo un numero positivo o negativo. Notate l'uso di C$ per cancellare i messaggi in caso di errore.

MODULO 6.2.10

```
18000 REM*******************************
18010 REM CANCELLAZIONI
18020 REM*******************************
18030 PL=I:IF LEFT$(A$(CD,PL),1)<>"*" TH
EN GR=1:GOTO 18060
18040 GR=0
18050 GR=GR+1:IF LEFT$(A$(CD,PL+GR),1)="
$" THEN GOTO 18050
18060 FOR K=PL TO C(CD)-GR-1:A(CD,K)=A(C
D,K+GR):A$(CD,K)=A$(CD,K+GR):NEXT
18070 C(CD)=C(CD)-GR:I=I-GR+1:RETURN
```

Questo modulo esegue le cancellazioni definite nel modulo precedente.

Commenti

Linea 18030: PL viene posto al valore della variabile di ciclo del modulo precedente. In caso di registrazioni singole e voci dettagliate, la variabile GR, che indica il numero degli elementi da cancellare, viene posta a uno.

Linea 18040: nel caso di voci principali, la variabile GR viene incrementata per tenere conto della voce principale stessa e di tutte le relative voci dettagliate, dal momento che queste vanno cancellate insieme con la voce principale.

Linea 18060: la variabile GR viene utilizzata per determinare quanti elementi andranno riscritti nel file e di quanto vada ridotto il valore del lato corrispondente della matrice C.

Collaudo dei moduli 6.2.8-6.2.10

A questo punto dovreste essere in grado di introdurre dei dati e di scorrerli, modificando gli importi associati o cancellando elementi a piacimento.

MODULO 6.2.11

```
19000 REM#******************************
19010 REM STAMPA
19020 REM*******************************
19030 LET PA$="CREDITO":IF CD=1 THEN PA$
="DEBITO"
19040 IT=0:SS=0:PRINT "
";PA$;""
19050 FOR I=0 TO C(CD)-1:IT=IT+A(CD,I)
19060 PRINT "";:IF I/2=INT(I/2) THEN PR
INT"";
19070 IF LEFT$(A$(CD,I),1)="*" THEN PRIN
T
19080 IF LEFT$(A$(CD,I),1)="$" THEN PRIN
T "";
19090 PRINTMID$(A$(CD,I),2)
19100 IF LEFT$(A$(CD,I),1)="*" THEN 1915
0
19110 PRINT "";
19120 IF LEFT$(A$(CD,I),1)="%" THEN PRIN
T "";
19130 MM=A(CD,I):GOSUB 21000:PRINTM$
19140 IF LEFT$(A$(CD,I),1)="$" THEN SS=S
S+A(CD,I)
19150 IF SS=0 OR LEFT$(A$(CD,I+1),1)="$"
 THEN 19180
19160 PRINT "------------
";
19170 MM=SS:GOSUB 21000:PRINT M$:SS=0
19180 GET GG$:IF GG$="" THEN 19180
19190 NEXT I:PRINT "
-----------"
19200 PRINT "TOTALE:
";
19210 MM=IT:GOSUB 21000:PRINTM$
19220 PRINT "PREMERE UN TASTO PER TERM
INARE"
19230 GET GG$:IF GG$="" THEN 19230
19240 RETURN
19250 STOP
```

Questo modulo è analogo a quello di stampa del programma precedente.

Commenti

Linea 19060: per assicurare la chiarezza nell'associare somme e importi, le voci e i relativi importi vengono stampati alternativamente in nero e verde.

Linea 19070: viene lasciata una linea bianca prima di stampare una voce principale.

Linea 19080: alle voci dettagliate vengono aggiunti due spazi.

Linee 19100-19140: vengono stampati i nomi e gli importi corrispondenti, per le voci dettagliate e per le registrazioni singole. Gli importi delle voci dettagliate vengono stampati in una colonna separata e i totali parziali di una stessa voce vengono cumulati in SS.

Linee 19150-19170: alla fine di un gruppo di voci dettagliate viene stampato il totale relativo a tutto il gruppo.

Linea 19180: anche in questo caso gli elementi vengono stampati uno per volta, in seguito all'azionamento di un tasto qualsiasi.

Linee 19200-19210: viene stampato il totale del lato pertinente del rendiconto.

Collaudo del modulo 6.2.11

Dopo aver introdotto alcuni dati, dovreste ora essere in grado di visualizzare ogni colonna dei rendiconti. Notate che è possibile visualizzare solo un lato alla volta.

Riepilogo

Giunti a questo punto, dovreste cominciare a prendere confidenza con le tecniche relative all'aggiunta e all'eliminazione di elementi da un archivio (file) senza pregiudicarne l'ordine complessivo. Dovreste inoltre aver imparato alcuni dei sottili trucchi che comporta la visualizzazione sullo schermo di numeri anche semplici in un formato ben congegnato. Prima di continuare vale comunque la pena di rivisitare alcuni dei metodi qui impiegati, dato che nel programma successivo dovremo trattare e visualizzare dati di complessità molto maggiore di quelli incontrati fino a questo punto.

Ulteriori sviluppi

1) Una utile funzione aggiuntiva potrebbe essere la possibilità di stampare il bilancio complessivo tra il dare e l'avere del rendiconto, una volta visualizzato ogni lato.
2) Come nel programma precedente, se avete intenzione di memorizzare numerosi elementi, desidererete modificare l'attuale modulo di ricerca, unicamente in grado di scorrere uno per uno i vari elementi. State attenti comunque nel fare ciò, dal momento che il modulo deve poter distinguere le voci principali mentre scorre l'archivio, in particolar modo per le cancellazioni. Saltare semplicemente da un posto all'altro nel file, senza tenere conto di questa necessità, potrebbe condurre a risultati disastrosi.

6.3 Bilancio preventivo

Rivolgiamo ora la nostra attenzione al programma più complesso e difficoltoso che incontrerete in questo libro. Bilancio preventivo è un ausilio finanziario potente e versatile che permette all'utente di pianificare le proprie finanze su un periodo di 12 mesi e di esaminare le conseguenze di decisioni particolarmente cruciali riguardo introiti e spese. Usato con discernimento, può fornire sorpren-

denti intuizioni sul bilancio familiare per l'anno entrante, oltre naturalmente ad illustrare alcuni dei problemi connessi con il trattamento di grandi quantità di dati numerici. Le matrici utilizzate dal programma contengono qualcosa come 800 valori numerici distinti.

Bilancio preventivo: lista delle variabili

Variabile	Descrizione
BA(1,11)	Bilancio cassa per ogni mese
BD(1,11)	Bilancio dei versamenti programmati sui versamenti effettivi
C1(1,11)	Introiti principali
C2(1,11)	Introiti supplementari
CU	Variabile temporanea utilizzata per calcolare il surplus/deficit cumulativo
FO$	Stringa di controllo cursore usata per formattare la tabella
H	Indicatore della colonna da indirizzare nelle matrici
I1	Variabile impiegata per assicurare una conveniente manipolazione dei periodi di 12 mesi che oltrepassano l'anno civile
M1	Variabile temporanea del mese per iniziare la visualizzazione della tabella
M2	Variabile temporanea che registra il cambiamento del mese corrente
MM	Numero del mese corrente
MO(1,29)	Versamento mensile medio per ogni voce
MO$(11)	Nomi dei mesi
MY	Variabile temporanea impiegata per formattare le cifre di denaro
MY$	Stringa contenente le cifre di denaro formattate
N(1)	Numero di elementi in entrambe le colonne delle matrici
PA(1,29,11)	Importi associati alle voci delle operazioni
PA$(1,29)	Nomi delle operazioni
PP	Variabile temporanea usata per indicare la posizione dell'elemento da modificare nella matrice
PT(1,11)	Totali mensili delle spese
R$	Separatore nel file dei dati
T(1)	Variabile temporanea impiegata per calcolare l'importo totale raggiunto nel versamento mensile medio
TT	Variabile temporanea impiegata per calcolare le spese totali per le voci incluse nel calcolo del bilancio medio
Y	Numero dell'ultimo mese dell'anno

MODULO 6.3.1

```
21000 REM#******************************
21010 REM FILE DATI
21020 REM*******************************
21030 PRINT "▒POSIZIONARE IL NASTRO, QUI
NDI PREMERE      ▒RETURN▒"
21040 INPUT "PER ARRESTO AUTOMATICO:";Q$
:POKE192,7:POKE1,39
21050 PRINT "▒▒FUNZIONI DISPONIBILI:"
```

```
21060 PRINT "█ 1)SALVATAGGIO DATI":PRINT
 " 2)CARICAMENTO DATI"
21070 INPUT "█FUNZIONE RICHIESTA:";Q:ON
Q GOTO 21080,21130:RETURN
21080 POKE 1,7:FOR I=1 TO 2000:NEXT
21090 OPEN 1,1,1,"BUDGET":PRINT#1,MM,R$,
Y:FOR H=0 TO 1:PRINT#1,N(H)
21100 FOR I=0 TO 11:PRINT#1,C1(H,I);R$;C
2(H,I):NEXT I
21110 FOR I=0 TO N(H)-1:IF PA$(H,I)="" T
HEN PA$(H,I)=" "
21120 PRINT#1,PA$(H,I):FOR J=0 TO 11:PRI
NT#1,PA(H,I,J):NEXT J,I,H:CLOSE1:RETURN
21130 OPEN 1,1,0,"BUDGET":INPUT#1,MM,Y:F
OR H=0 TO 1:INPUT#1,N(H)
21140 FOR I=0 TO 11:INPUT#1,C1(H,I),C2(H
,I):NEXT I
21150 FOR I=0 TO N(H)-1
21160 INPUT#1,PA$(H,I):FOR J=0 TO 11:INP
UT#1,PA(H,I,J):NEXT J,I
21170 GOSUB 14000:NEXT H:CLOSE1:RETURN
```

La complessità di questo modulo di file dei dati dovrebbe convincervi della necessità di eseguire un salvataggio a intervalli regolari per mettervi al riparo dagli errori, inevitabili nell'introduzione di un programma complesso come questo.

MODULO 6.3.2

```
11000 REM#**********************************
11010 REM MENU
11020 REM**********************************
11030 POKE53281,13:PRINT "█████████████BI
LANCIO FAMILIARE"
11040 PRINT "███FUNZIONI DISPONIBILI:"
11050 PRINT " ██1)VISUALIZZA ANALISI MENS
ILE"
11060 PRINT " 2)VARIAZIONI"
11070 PRINT " 3)NUOVE INTESTAZIONI"
11080 PRINT " 4)CANCELLAZIONE INTESTAZIO
NI"
11090 PRINT " 5)AZZERAMENTO IPOTESI"
11100 PRINT " 6)AGGIORNAMENTO MESE"
11110 PRINT " 7)ARCHIVIAZIONE DATI"
11120 PRINT " 8)INIZIALIZZAZIONE"
11130 PRINT " 9)TERMINE"
11140 INPUT "███FUNZIONE RICHIESTA:";Z:PR
INT "█";:IF Z<5 THEN 11160
11150 ON Z-4 GOSUB 15000,17000,21000,120
00,11190:GOTO 11000
11160 PRINT "█████████████1)DATI REALI":PR
INT "2)DATI IPOTETICI"
```

```
11170 INPUT "■FUNZIONE RICHIESTA:";H:IF
H<1 OR H>2 THEN 11170
11180 PRINT "■":H=H-1:ON Z GOSUB 13000,1
9000,16000,20000:GOTO 11000
11190 PRINT "■■■■■■■■■■■■■■■■■■■■■■■■B
ILANCIO CHIUSO":END
```

Un normale modulo di menu, con l'aggiunta della possibilità di definire se sta per essere indirizzata la colonna reale delle matrici o quella ipotetica. La distinzione verrà spiegata nel seguito.

MODULO 6.3.3

```
12000 REM#*******************************
12010 REM INIZIALIZZAZIONE
12020 REM********************************
12030 CLR
12040 DIM PA$(1,29),MO(1,29),PA(1,29,11)
,PT(1,11),BD(1,11),C1(1,11),BA(1,11)
12050 DIM C2(1,11):R$=CHR$(13)
12060 DATA GENNAIO,FEBBRAIO,MARZO,APRILE
,MAGGIO,GIUGNO,LUGLIO
12070 DATA AGOSTO,SETTEMBRE,OTTOBRE,NOVE
MBRE,DICEMBRE
12080 DIM MO$(11):RESTORE:FOR I=0 TO 11:
READ MO$(I):NEXT
12090 INPUT "■■CARICAMENTO DAL NASTRO (S
/N):";Q$:IF Q$="S" THEN GOTO 11000
12100 INPUT "■■INTRODURRE IL NUMERO DEL
MESE CORRENTE";MM:MM=MM-1:Y=MM+11
12110 GOSUB 18000:GOSUB 16000:GOTO 11000
```

Modulo d'inizializzazione.

MODULO 6.3.4

```
18000 REM#*******************************
18010 REM INTROITI
18020 REM********************************
18030 PRINT "■■■INTRODURRE GLI INTROITI
PRINCIPALI COME SEGUE:"
18040 FOR I=MM TO Y: I1=I:IF I1>11 THEN
I1=I1-12
18050 PRINT MO$(I1);":":INPUT "■■■■■■■■■
■■■■";C1(H,I1):NEXT I
18060 PRINT "■■ALTRI INTROITI PREVISTI:"
18070 FOR I=MM TO Y:I1=I:IF I>11 THEN I1
=I1-12
18080 PRINT MO$(I1);":":INPUT "■■■■■■■■■
■■■■";C2(H,I1):NEXT I
18090 GOSUB 14000:RETURN
```

Questo modulo permette l'introduzione degli introiti, sotto le intestazioni di introiti principali e introiti supplementari.

Commenti

Linea 18040: benché i dati siano memorizzati nelle matrici nell'ordine Gennaio-Dicembre, il periodo di 12 mesi che il programma è in grado di coprire può iniziare in qualsiasi mese. Pertanto si utilizza la variabile I1 per assicurarsi che, completato il dodicesimo mese, il ciclo continui per indirizzare il primo mese dell'anno.

Linea 18050: notate il modo di utilizzare la variabile H per determinare quale colonna delle matrici venga indirizzata.

Collaudo dei moduli 6.3.2-6.3.4

Inserendo dei RETURN temporanei a 14000 e a 16000 dovreste poter inserire i dati relativi agli introiti per i 12 mesi seguenti il mese d'inizio da voi scelto. Per il momento continuate a inserire dati nella colonna reale delle matrici — si chiarirà tutto in seguito.

MODULO 6.3.5

```
16000 REM******************************
16010 REM INTRODUZIONE DEI PAGAMENTI
16020 REM******************************
16030 PRINT "███████████████INTRODUZIONE FA
TTURE"
16040 PRINT "█ANTEPORRE '*' A UN NOME":P
RINT "DA NON INSERIRE IN BILANCIO"
16050 PRINT "██('ZZZ' PER TERMINARE)"
16060 INPUT "██INTESTAZIONE DELLA FATTUR
A:";Q$:IFQ$="ZZZ"THENGOSUB 14000:RETURN
16070 N(H)=N(H)+1
16080 IF N(H)=30 THEN N(H)=29:PRINT "SPA
ZIO ESAURITO":FORI=0TO2000:NEXT:RETURN
16090 PA$(H,N(H)-1)=Q$:PRINT"██VERSAMENT
I PER ";Q$;":█"
16100 FOR I=MM TO Y:I1=I:IF I1>11 THEN I
1=I1-12
16110 PRINT MO$(I1):INPUT "████████████"
;PA(H,N(H)-1,I1):NEXT I
16120 GOTO 16000
```

Questo modulo permette d'introdurre le intestazioni delle fatture e i relativi importi.

Commenti

Linea 16040: il programma è in grado di calcolare una cifra media mensile che coprirà il totale dell'esborso annuale sotto ciascuna intestazione. Premettendo al nome un * si esclude dal procedimento quel particolare pagamento — viene cioè trattato come una voce saltuaria.

Linea 16070: si utilizza la variabile H per incrementare l'una o l'altra posizione della matrice N a due elementi, che registra il numero di operazioni memorizzate in ogni colonna di PA$.

Linee 16090-16110: dopo avere specificato l'intestazione del pagamento, si domanda d'inserire l'ammontare per ciascuno dei 12 mesi del periodo coperto.

Collaudo del modulo 6.3.5

Dovreste ora poter introdurre alcune fatture, per ritrovarle memorizzate nella colonna zero delle matrici PA$ e PA — rimanendo nelle colonne reali delle matrici. Per questa prova è necessario mantenere il RETURN temporaneo a 14000.

MODULO 6.3.6

```
14000 REM#*********************************
14010 REM AGGIORNAMENTI
14020 REM**********************************
14030 T(H)=0
14040 FOR I=0 TO N(H)-1:BU=0:IF LEFT$(PA
$(H,I),1)="*" THEN 14060
14050 FOR J=0 TO 11:BU=BU+PA(H,I,J):NEXT
:MO(H,I)=BU/12:T(H)=T(H)+MO(H,I)
14060 NEXT I
14070 TT=0:CU=0:FOR I=MM TO Y:I1=I+12*(I
>11):PT(H,I1)=0
14080 FOR J=0 TO N(H)-1:PT(H,I1)=PT(H,I1
)+PA(H,J,I1):NEXT J:TT=TT+PT(H,I1)
14090 FOR J=0 TO N(H)-1:IF LEFT$(PA$(H,J
),1)="*" THEN TT=TT-PA(H,J,I1):NEXT J
14100 BD(H,I1)=T(H)*(I-MM+1)-TT:CU=CU+C1
(H,I1)+C2(H,I1)-PT(H,I1):BA(H,I1)=CU
14110 NEXT I:RETURN
```

Questo modulo esegue tutti i calcoli necessari per la costruzione della tabella delle cifre per cui stiamo lavorando.

Commenti

Linee 14040-14060: le cifre del preventivo medio mensile vengono calcolate e memorizzate nella matrice MO. In T viene memorizzato il totale cumulativo di queste cifre. Il procedimento non viene eseguito per le intestazioni di pagamento che iniziano con un *.

Linea 14070: notate l'uso della condizione logica (I > 11) per calcolare il valore di I1. Se I è minore o uguale a 11, la condizione varrà zero e non influenzerà il valore di I1. Quando I è maggiore di 11, la condizione assumerà il valore di meno uno e si può utilizzare per riposizionare I1.

Linea 14080: i totali di tutte le fatture da pagare in un dato mese vengono sommati nella linea pertinente di PT. TT viene utilizzato per contenere il totale generale di tutti questi totali mensili.

Linea 14090: le somme associate a qualsiasi voce che non deve essere inclusa nei calcoli del preventivo medio vengono ora sottratte da TT, che a questo punto contiene il totale generale delle voci incluse nel computo del preventivo medio.

Linea 14100: viene ora memorizzato nella matrice BD il bilancio delle cifre preventivate contro i pagamenti effettivi, moltiplicando l'esborso medio mensile per il numero dei mesi e sottraendo i pagamenti effettuati sulle voci di preventivo fino al mese considerato. Nella ma-

trice BA si pone il bilancio tra i due tipi d'introito e il totale dei pagamenti effettuati durante il mese.

Collaudo del modulo 6.3.6

Non è facile collaudare completamente questo modulo finché non si è inserito quello di visualizzazione, tuttavia è cosa buona introdurre alcuni dati, dal momento che ciò richiamerà questo modulo e controllerà la sintassi. Se confidate nel buon funzionamento del modulo, è conveniente procedere al salvataggio su nastro dei dati introdotti.

MODULO 6.3.7

```
22000 REM#********************************
22010 REM ROUTINE  FUNZIONALE
22020 REM*********************************
22030 MJ=MY/1000
22040 MJ=INT(ABS(MJ)+10000):MY$=MID$(STR
$(MJ),3):IF MJ>=20000 THEN MY$="####"
22050 RETURN
```

Una routine di formattazione che restituisce un numero di quattro cifre, completo di zeri non significativi, se necessario. Le cifre stampate si riferiranno alle migliaia di lire. Inoltre, se il dato da stampare è maggiore di £ 9999000, verrà visualizzato '####', per indicare il fatto che eccede le capacità di visualizzazione del programma. I calcoli effettuati non ne risentiranno comunque.

MODULO 6.3.8

```
13000 REM#********************************
13010 REM VISUALIZZAZIONE
13020 REM*********************************
13030 PRINT "TACCUINO"
13040 INPUT"NUMERO DEL MESE INIZIALE";M1
:IF M1<1 OR M1>11 THEN 13040
13050 M1=M1-1:IF MM-M1-12*(M1>MM-1)<4 TH
EN M1=MM-4-12*(MM<5)
13060 PRINT "
"
13070 PRINT " MESE          ";
13080 FORJ=M1 TO M1+3:PRINT "";LEFT$
(MO$(J+12*(J>11)),3);
13090 NEXT J:PRINT " B "
13100 PRINT "
"
13110 FOR I=0 TO N(H)-1:IF I<>15 THEN 13
140
13120 INPUT "RETURNPER PULIRE LO SCHE
RMO E CONTINUARE";Q$:PRINT ""
13125 FOR J=1 TO 20
13130 PRINT "
          ":NEXT J:PRINT ""
13140 PRINT " ";LEFT$(PA$(H,I),12):PR
INT "";
```

```
13150 FOR J=M1 TO M1+3:PRINT "▒▒▒";:MY=I
NT(PA(H,I,J+12*(J>11))):GOSUB 22030
13160 PRINT MY$;:NEXT J:PRINT "▒▒▒";
13170 MY=INT(MO(H,I)):GOSUB 22030:PRINT
MY$:NEXT I
13180 PRINT " ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
13190 INPUT "■PREMERE ▒RETURN■ PER L'ANA
LISI";Q$
13200 PRINT"▒▒▒▒":FORI=0TO20:PRINT"
                                      "
13210 NEXT I:RESTORE:FOR J=1 TO 12:READ
A$:NEXT:PRINT "▒▒▒▒▒"
13220 DATA TOTALE,BUDGET,BIL PREV,INCASS
I PRINC,INCASSI SUPPL,INCASSI TOT
13230 DATA BIL CASSA,BIL COM
13240 FOR I=1 TO 8:READ A$:PRINT " ▒▒▒";
A$
13250 PRINT " ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒":NEXT I
13260 FO$="▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
13270 FOR I=M1 TO M1+3:I1=I+12*(I>11):PR
INT "▒▒▒▒"
13280 PRINT FO$;"▒▒";:MY=PT(H,I1):PRINT
"■";:IF MY<0 THEN PRINT "▒";
13290 GOSUB 22030:PRINT MY$:PRINT
13300 PRINT FO$;"▒▒";:MY=T(H):PRINT "■";
:IF MY<0 THEN PRINT "▒";
13310 GOSUB 22030:PRINT MY$:PRINT
13320 PRINT FO$;"▒▒";:MY=BD(H,I1):PRINT
"■";:IF MY<0 THEN PRINT "▒";
13330 GOSUB 22030:PRINT MY$:PRINT
13340 PRINT FO$;"▒▒";:MY=C1(H,I1):PRINT
"■";:IF MY<0 THEN PRINT "▒";
13350 GOSUB 22030:PRINT MY$:PRINT
13360 PRINT FO$;"▒▒";:MY=C2(H,I1):PRINT
"■";:IF MY<0 THEN PRINT "▒";
13370 GOSUB 22030:PRINT MY$:PRINT
13380 PRINT FO$;"▒▒";:MY=MY+C1(H,I1):PRI
NT "■";:IF MY<0 THEN PRINT "▒";
13390 GOSUB 22030:PRINT MY$:PRINT
13400 PRINT FO$;"▒▒";:MY=MY-PT(H,I1):PRI
NT "■";:IF MY<0 THEN PRINT "▒";
13410 GOSUB 22030:PRINT MY$:PRINT
13420 PRINT FO$;"▒▒";:MY=BA(H,I1):PRINT
"■";:IF MY<0 THEN PRINT "▒";
13430 GOSUB 22030:PRINT MY$:PRINT
13440 FO$=FO$+"▒▒▒▒▒":NEXT I
13450 INPUT "■RIPETERE VISUALIZZAZIONE (
S/N) ";Q$
```

```
13460 IF Q$="S" THEN 13060
13470 RETURN
```

Nel programma precedente abbiamo constatato che i moduli di visualizzazione sono spesso la parte più complessa di un programma il cui scopo sia presentare una tabella di dati, e anche questo non fa certamente eccezione. Ciò detto, va notato che sotto l'apparente complessità questo è un modulo relativamente semplice, che raccoglie cifre già calcolate e le pone sullo schermo. Appare complesso solo a causa del numero di cifre da stampare.

Commenti

Linee 13040-13050: la tabella visualizza le cifre relative a quattro mesi successivi a quello indicato dall'utente. Tuttavia oltrepassare il termine del periodo di 12 mesi corrente toglierebbe significato alla tabella, pertanto, se viene introdotta una cifra corrispondente a un periodo inferiore ai quattro mesi dal termine, il mese iniziale viene riposizionato.

Linee 13060-13100: si stampano l'intestazione della tabella, consistente nelle prime tre lettere del mese considerato, e una intestazione per la colonna ''bilancio medio''.

Linee 13100-13170: i nomi corrispondenti ai pagamenti vengono ottenuti dalla matrice PA$ e stampati nella colonna a sinistra. Di seguito, le cifre relative ai quattro mesi e la cifra del bilancio medio per ogni voce vengono stampate sullo schermo in senso orizzontale, separate da caratteri grafici nelle colonne, utilizzando il modulo precedente per formattare gli importi, con gli zeri non significativi se necessario. Vengono stampate 15 linee, tenendo conto della possibilità di pulire lo schermo e stamparne altrettante se ciò non fosse sufficiente. Il programma può trattare fino a 30 intestazioni.

Linee 13210-13240: i titoli per le cifre nella seconda parte della tabella vengono letti dalle frasi DATA e stampati verticalmente lungo il lato destro dello schermo — l'intestazione della tabella non viene modificata (l'intestazione della colonna di bilancio è ora in sovrappiù, ma non viene cancellata).

Linee 13260-13440: nonostante la lunghezza, si tratta di una routine semplice che, usando la stringa FO$ per determinare la posizione della colonna, stampa in senso verticale le cifre corrispondenti a ogni mese di fronte alle proprie intestazioni. Al termine della colonna di ogni mese, vengono aggiunti a FO$ cinque caratteri di spostamento a destra del cursore e si ripete il procedimento in una nuova colonna, per il mese successivo. Notate l'uso dei caratteri di controllo rosso e nero per indicare se una voce sia positiva o negativa.

Collaudo dei moduli 6.3.7 e 6.3.8

Avendo dei dati già memorizzati, dovreste ora essere in grado di visualizzarli sullo schermo. Per controllare la tabella (a parte il suo aspetto esteriore) è necessario comprendere il significato delle varie cifre visualizzate.

TOTALE — È il totale di tutti i pagamenti da eseguire nel mese

PREVENTIVO — Lo stesso per ogni mese, è questa la somma media da accantonare al fine di coprire, nel periodo di 12 mesi considerato, tutte le fatture non saldate. Un preventivo medio non necessariamente coprirà tutte le uscite di ogni singolo mese (per esempio se tutti i pagamenti sono stati effettuati durante il primo mese). Questa cifra segnala se la somma accantonata in media si trovi al passo con gli effettivi pagamenti che è chiamata a coprire. Alla fine del periodo di 12 mesi sarà zero

BILANCIO CASSA — Differenza tra introiti e spese nel mese considerato

BILANCIO COMPL. — Differenza tra introiti totali e spese totali dall'inizio del periodo di 12 mesi

Le voci INTROITI PRINCIPALI/INTROITI SUPP./INTROITI TOTALI non necessitano di particolari spiegazioni.

Notate come si avranno delle piccole discrepanze, dal momento che vengono visualizzate solo cifre intere (in multipli di 1000 lire), mentre i calcoli vengono effettivamente compiuti su cifre complete. Perciò il preventivo mensile per una spesa di 37(000) lire sarà visualizzato come 3(000) lire, ma ciò non influenzerà il calcolo esatto del preventivo mensile totale.

MODULO 6.3.9

```
19000 REM#*****************************
19010 REM MODIFICHE
19020 REM******************************
19030 PRINT "MODIFICHE
"
19040 PRINT "FUNZIONI DISPONIBILI:MODI
FICA"
19050 PRINT " 1)VOCI DI SPESA"
19060 PRINT " 2)INCASSI PRINCIPALI"
19070 PRINT " 3)INCASSI SUPPLEMENTARI"
19080 INPUT "FUNZIONE RICHIESTA:";QQ:O
N QQ GOSUB 19100,19190,19190
19090 GOSUB 14000:RETURN
19100 INPUT "INTESTAZIONE DA MODIFICARE:
";Q$
19110 FOR I=0 TO N(H)-1:IF Q$<>PA$(H,I)
THEN 19130
19120 PP=I:GOTO 19140
19130 NEXT I:PRINT "VOCE NON PRESENTE":
FOR I=1TO2000:NEXTI:RETURN
19140 PRINT"";PA$(H,PP):PRINT"INTROD
URRE NUOVO VALORE O '*' PER CONFERMARE"
19150 FOR I=MM TO Y:I1=I+12*(I>11)
19160 PRINT MO$(I1):PRINT "
:";PA(H,PP,I1);:INPUT Q$
```

```
19170 IF Q$<>"*" THEN PA(H,PP,I1)=VAL(Q$
)
19180 NEXT I:RETURN
19190 IF QQ=2 THEN PRINT "INCASSI PRI
NCIPALI:";
19200 IF QQ=3 THEN PRINT "INCASSI SUP
PLEMENTARI:";
19205 PRINT " '*' PER CONFERMARE "
19210 FOR I=MM TO Y:I1=I+12*(I>11)
19220 PRINT MO$(I1):PRINT "
:";
19230 IF QQ=2 THEN PRINT C1(H,I1);
19240 IF QQ=3 THEN PRINT C2(H,I1);
19250 INPUT Q$:IF Q$<>"*" AND QQ=2 THEN
C1(H,I1)=VAL(Q$)
19260 IF Q$<>"*" AND QQ=3 THEN C2(H,I1)=
VAL(Q$)
19270 NEXT I:RETURN
```

Nel caso fosse necessaria una modifica a una voce già inserita, questo modulo permetterà all'utente di specificare se si tratti di una voce di spesa degli introiti principali o di quelli supplementari. Vengono quindi visualizzate le cifre corrispondenti e l'utente può sia confermarle (battendo un *) sia introdurre un nuovo valore.

MODULO 6.3.10

```
20000 REM#################################
20010 REM CANCELLA INTESTAZIONI
20020 REM#################################
20030 INPUT "VOCE DA CANCELLARE:";Q$
20050 FOR I=0 TO N(H)-1:IF Q$=PA$(H,I) T
HEN 20080
20070 NEXT I:PRINT "VOCE NON PRESENTE"
:FOR J=1 TO 2000:NEXT J:RETURN
20080 N(H)=N(H)-1:FOR J=1 TO N(H)-1:PA$(
H,J)=PA$(H,J+1)
20090 FOR K=0 TO 11:PA(H,J,K)=PA(H,J+1,K
):NEXT K,J:GOSUB 14000:RETURN
```

Questo modulo permette di cancellare una intestazione qualsiasi.

MODULO 6.3.11

```
17000 REM#################################
17010 REM REGISTRA MESE
17020 REM#################################
17030 PRINT "AGGIORNAMENTO
MESE"
17040 INPUT"INTRODURRE IL NUMERO DEL M
ESE CORRENTE:";M2:IFM2<0ORM2>12THEN17040
17050 M2=M2-1:IF M2=MM THEN RETURN
17060 IF M2<MM THEN M2=M2+12
17070 FOR I=MM TO M2:I1=I+12*(I>11)
```

```
17080 PRINT "AGGIORNAMENTO
MESE"
17090 PRINT"INTRODURRE COMLETAMENTE GL
I IMPORTI PER IL PROSSIMO";MO$(I1);":"
17100 FOR J=0 TO N(0)-1:PRINTPA$(0,J);"
(";PA(0,J,I1);"):";:INPUT PA(0,J,I1)
17110 NEXT J
17120 INPUT "INTROITI PRINCIPALI:";C1(
0,I1)
17130 INPUT "INTROITI SUPPLEMENTARI:";
C2(0,I1):NEXT I
17140 MM=M2+12*(M2>11):Y=MM+11:H=0:GOSUB
 14000:GOSUB 15000:RETURN
```

Lo scopo di questo modulo è permettere cambiamenti di mese. Quando l'utente specifica che è cambiato il mese corrente, vengono richiesti nuovi dati per ogni voce di spesa e tipo d'introito, per ciascuno dei mesi trascorsi, e devono ora essere aggiunti al termine del periodo di 12 mesi. Così, se il periodo precedente iniziava con Maggio, mentre quello nuovo inizia con Luglio, all'utente verranno richieste solo le cifre corrispondenti a Maggio e Giugno, dal momento che questi diventano ora gli ultimi due mesi del periodo di 12.

Collaudo dei moduli 6.3.9-6.3.11

A questo punto dovreste poter modificare le cifre delle voci di spesa o degli introiti, cancellare delle voci di spesa e modificare il periodo che il programma ricopre. Per collaudare l'ultimo modulo dovrete introdurre un RETURN temporaneo alla linea 15000.

```
15000 REM#********************************
15010 REM IPOTESI
15020 REM*********************************
15030 T(1)=T(0)
15040 FOR I=0 TO N(0)-1:PA$(1,I)=PA$(0,I
):MO(1,I)=MO(0,I)
15050 FOR J=0 TO 11:PA(1,I,J)=PA(0,I,J):
NEXT J,I
15060 FOR J=0 TO 11:PT(1,J)=PT(0,J):BD(1
,J)=BD(0,J):C1(1,J)=C1(0,J)
15070 BA(1,J)=BA(0,J):C2(1,J)=C2(0,J):NE
XT J:N(1)=N(0):RETURN
```

MODULO 6.3.12

Questo semplice modulo è uno dei più importanti del programma. Ciò che fa è copiare i dati inseriti nella colonna ''reale'' delle matrici in quella ''ipotetica''. Uno dei principali punti di forza di questo programma è poter scegliere di introdurre dei dati nella colonna delle matrici riservata alle ipotesi, per provare gli effetti di una decisione finanziaria senza che ciò sortisca alcun effetto sui dati reali.
Sui dati ipotetici si possono effettuare tutte le operazioni del programma e, quando sarete soddisfatti del vostro operato, tutto ciò che dovrete fare è richiamare questo modulo: i dati delle colonne

ipotetiche verranno istantaneamente riportati a quelli reali. Questo modulo viene richiamato automaticamente quando viene modificato il mese, altrimenti i due lati delle tabelle potrebbero lavorare su periodi differenti.

Collaudo del modulo 6.3.12

In effetti ora siete in grado di collaudare gli aspetti ipotetici di tutte le funzioni semplicemente specificando dati ipotetici quando le funzioni vengono richiamate. Aggiungete e togliete voci da entrambi i lati, usate poi la visualizzazione della tabella per verificare che nessuno dei due lati influenzi l'altro. Usate poi questo modulo per copiare i dati reali nella parte di quelli ipotetici. Notate che la parte ipotetica è vuota, all'inizializzazione del programma.
Se le funzioni ipotetiche operano correttamente, il programma è pronto per l'uso.

Riepilogo

Questo lungo programma, usato in modo corretto, è di notevole potenza, benché richieda un po' di esercizio per poterne trarre il meglio. Preso seriamente, può darvi alcune sorprendenti informazioni sullo stato delle vostre finanze durante l'anno — quando si avranno delle ristrettezze e quando si potrà invece largheggiare; come eventualmente spostare i pagamenti per avere da parte qualcosa per Natale o per le vacanze; quale potrebbe essere l'effetto complessivo di un investimento o di un aumento della rendita.
Ricordate tuttavia che questo libro è pensato per mettere il 64 al lavoro in vece vostra. Se avete positivamente superato i problemi sollevati dal debugging di questo programma, non ci sarà alcun motivo per non proseguire nel tentativo di utilizzarlo in altre situazioni che richiedano una versatile introduzione e manipolazione di dati, oltre ad una chiara presentazione sotto forma di tabelle e alla possibilità di elaborazione di due insiemi di dati nello stesso tempo. Il programma si può considerare come un punto di partenza per mettere al lavoro il 64 e le vostre nuove capacità.

Ulteriori sviluppi

1) Il programma potrebbe risultare più utile se aveste la possibilità di copiare le matrici dei dati ipotetici in quelle reali. Ciò dovrebbe comportare solamente una piccola modifica a un modulo.
2) Riduzioni nella lunghezza del programma possono derivare dalla diminuzione del numero di matrici, comprimendo la stessa quantità di dati in matrici meno numerose ma più complesse. Dovreste quindi poter stampare i dati tramite pochi cicli.
3) Volendo modificare solo un valore di un pagamento o di un introito dovete attraversare tutti i dodici pagamenti. Provate ad aggiungere la possibilità di saltare al periodo considerato e di uscirne una volta completata la modifica che volevate apportare.

7. Musica

Una delle meraviglie del 64 è il modo in cui la qualità e l'estrema duttilità delle sue capacità sonore dischiudono al possessore di un micro un nuovo mondo di possibilità. In un futuro non troppo lontano verranno senza dubbio scritti libri sull'uso del chip d'interfaccia sonora del 64 (SID = *Sound Interface Device* = dispositivo d'interfaccia sonora).
L'incredibile complessità delle possibilità del chip SID fa sì che nessun programma possa render loro piena giustizia e un capitolo di un'opera di carattere generale non può servire altro che da introduzione verso le pressoché infinite combinazioni di suoni disponibili. Ciò detto, tuttavia, il programma qui presentato è tra quelli che forniscono delle solide fondamenta per futuri esperimenti e creazioni. Lo scopo del programma non è solo mettere l'utente in grado di definire e suonare delle note (cosa che peraltro fa), ma anche rendere direttamente accessibile all'utente ogni parte del SID. La maggior parte di ciò che il SID può ottenere è messa a disposizione abbastanza semplicemente, usando questo programma come tramite.
La prima cosa da ricordare è che una nota musicale non è semplicemente una vibrazione di una determinata frequenza: in realtà è una combinazione di diverse frequenze, alte e basse. Ognuna delle tre voci del SID richiede l'introduzione del valore di frequenza della nota richiesta e, a questo scopo, è necessario assegnare un valore a due byte (o, più precisamente, a un byte "basso" e a un byte "alto"). Il programma deve poter accettare delle note in una forma comprensibile all'utente, per poi tradurle in una forma utilizzabile dal SID. In secondo luogo, l'intensità di ogni singola nota varia in modo complesso mentre viene eseguita:
a) la prima fase è nota come l'ATTACCO. È questa la velocità con cui la nota si porta dal silenzio al suo valore di picco. Tanto più breve è il periodo di tempo dell'attacco, tanto più stridula risulterà la nota;
b) la seconda fase è detta DECADIMENTO. Durante questa fase la nota scende dal suo valore iniziale di picco;
c) dopo questa prima discesa, la nota entra nella fase di SUSTAIN

(sostenere), che determina la lunghezza del corpo principale della nota;

d) alla fine la nota si affievolisce nella fase del RILASCIO, che, come l'ATTACCO, può essere brusco o graduale.

Strumenti musicali diversi producono note di diverse caratteristiche, abbastanza indipendentemente dalla nota suonata e dal modo in cui varia la sua intensità. Queste differenze dipendono dalla forma d'onda del suono prodotto dallo strumento.
Il SID fa sì che ognuna delle sue tre voci possa produrre una qualsiasi delle tre forme d'onda musicali e un segnale di rumore bianco che è utile nella creazione di effetti sonori.
Una delle tre forme d'onda, quella impulsiva, è a sua volta capace di una notevole gamma di variazioni, modificando la lunghezza degli impulsi stessi.
Ottenute alla fine frequenza, tonalità e forma desiderate, il chip SID permette il filtraggio delle note. Ciò significa che, all'interno della nota, diverse frequenze possono essere ridotte d'intensità, mentre altre non vengono variate.

Musica: lista delle variabili

FI%(3)	Caratteristiche di filtraggio delle tre diverse voci
HF%(2,1000)	Valori alti delle frequenze di ogni nota nel motivo da suonare
IN	Puntatore di inizializzazione
LF%(2,1000)	Valori bassi delle frequenze di ogni nota nel motivo da suonare
R$	Separatore per i file dei dati
NL	Lunghezza della nota
NO%(1,95)	Valori alti e bassi delle frequenze delle 96 note disponibili
NT	Valore della nota ricavato dall'Appendice M del *Manuale d'utente.*
VO%(2,6)	Valori definibili da forzare tramite POKE nel chip SID per determinare le caratteristiche del suono delle tre voci
VS	Indirizzo di partenza di una voce nel SID.
WF%(2,1000)	Valori delle forme d'onda di ogni nota da suonare
WW	Valore della forma d'onda per ogni singola nota

MODULO 7.1.1

```
11000 REM#******************************
11010 REM MENU
11020 REM*******************************
11030 POKE 53281,15:PRINT "MUSICA"
11040 PRINT "FUNZIONI DISPONIBILI:"
11050 PRINT "1)PREDISPOSIZIONE VOCE"
11060 PRINT "2)ESECUZIONE MOTIVO CORREN
TE"
11070 PRINT "3)COMPILAZIONE MOTIVO"
11080 PRINT "4)ARCHIVIAZIONE DATI"
11090 PRINT "5)INIZIALIZZAZIONE"
```

```
11100 PRINT "■6)TERMINE"
11110 INPUT "■■FUNZIONE RICHIESTA:";Z:PR
INT "■";
11120 IFIN=0AND(Z<5)THENPRINT"■NON INIZI
ALIZZATO":FORI=1TO2000:NEXT:GOTO11000
11130 ON Z GOSUB 15000,14000,13000,17000
,12000,11140:GOTO11000
11140 PRINT "■■■■■■■■■■■■■■■■■■■■■■■MUS
ICA TERMINATA":END
```

Un normale modulo di menu.

MODULO 7.1.2

```
18000 REM#################################
18010 REM DATI TABELLA NOTE
18020 REM*********************************
18030 REM FREQUENZE NOTE
18040 DATA 269,284,301,318,337,358,379,4
01,425
18045 DATA451,477,506
18050 DATA536,568,602,637,675,716,758,80
3
18055 DATA851,902,955,1012
18060 DATA 1072,1136,1204,1275,1351,1432
,1517,1607,1703,1804,1911,2025
18070 DATA 2145,2273,2408,2551,2703,2864
,3034,3215,3406,3608,3823,4050
18080 DATA 4291,4546,4817,5103,5407,5728
,6069,6430,6812,7217,7647,8101
18090 DATA 8583,9093,9634,10207,10814,11
457,12138,12860,13625,14435,15294,16203
18100 DATA 17167,18187,19269,20415,21629
,22915,24277,25721,27250,28871,30588
18110 DATA 32407
18120 DATA 34334,36376,38539,40830,43258
,45830,48556,51443,54502,57743,61176
18130 DATA 64814
```

I dati di questa tabella servono semplicemente per introdurre i valori delle frequenze delle note. Ogni numero rappresenta il valore della frequenza riprodotta, moltiplicato per il numero fisso 16.401.

MODULO 7.1.3

```
12000 REM#################################
12010 REM COSTRUZIONE TABELLE
12020 REM*********************************
12030 CLR:DIM NO%(1,95):FOR I=0 TO 95:RE
AD NN:NO%(0,I)=INT(NN/256)
12040 NO%(1,I)=NN-256*INT(NN/256):NEXT
12050 DIM VO%(2,6),FI%(3),LF%(2,1000),HF
%(2,1000),WF%(2,1000)
12070 IN=1:R$=CHR$(13)
12080 GOTO 11000
```

L'uso delle variabili principali qui definite viene spiegato nella lista delle variabili.

Commenti

Linee 12030-12040: vengono letti e decodificati i valori alti e bassi delle frequenze per ognuna delle 95 note. Non possono essere memorizzati nella matrice sotto forma di numero singolo, dal momento che la matrice è definita come formata da numeri interi e può contenere numeri solo fino a 32767. I valori alti e quelli bassi vengono rispettivamente posti in NO%(0) e NO%(1).

Collaudo del modulo 7.1.3

Dopo aver richiamato questo modulo, dovreste poter leggere nella tabella valori alti e bassi di frequenze, approssimativamente uguali a quelli dell'Appendice M del *Manuale d'utente*. Notate che non saranno identici, dal momento che i valori qui utilizzati sono presi dalla *Guida di riferimento del programmatore*, la cui tabella presenta alcune differenze.

MODULO 7.1.4

```
15000 REM#********************************
15010 REM PREDISPOSIZIONI VOCI
15020 REM*********************************
15030 INPUT "[illegible]NUMERO DELLA VOCE(1-3):"
;V:IF V<1 OR V>3 THEN 15030
15040 VS=54272+7*(V-1)
15050 PRINT "[illegible]VOCE";V
15060 REM*********************************
15070 T1$="LARGHEZZA IMPULSO F/O (BASSO:
0-255):"
15080 PRINT "■";T1$;VO%(V-1,2);:Q$=""
15090 INPUT Q$:IF Q$<>"" THEN VO%(V-1,2)
=VAL(Q$)
15100 REM*********************************
15110 T1$="LARGHEZZA IMPULSO F/O (ALTO:0
-15):"
15120 PRINT T1$;VO%(V-1,3) AND 15;:Q$=""
15130 INPUT Q$:IF Q$<>"" THEN VO%(V-1,3)
=VAL(Q$)
15140 REM*********************************
15150 T1$="RUMORE BIANCO (1=SI/0=NO):"
15160 PRINT T1$;(VO%(V-1,4) AND 128)/128
;:Q$=""
15170 INPUT Q$:IF Q$<>"" THEN VO%(V-1,4)
=(VO%(V-1,4) AND 126)OR(VAL(Q$)*129)
15180 REM*********************************
15190 T1$="F/O IMPULSIVA (1=SI/0=NO):"
15200 PRINT T1$;(VO%(V-1,4) AND 64)/64;:
Q$=""
15210 INPUT Q$:IF Q$<>"" THEN VO%(V-1,4)
=(VO%(V-1,4) AND 190)OR(VAL(Q$)*65)
15220 REM*********************************
15230 T1$="F/O RAMPA (1=SI/0=NO):"
```

```
15240 PRINT T1$;(VO%(V-1,4) AND 32)/32;:
Q$=""
15250 INPUT Q$:IF Q$<>"" THEN VO%(V-1,4)
=(VO%(V-1,4) AND 222)OR(VAL(Q$)*33)
15260 REM******************************
15270 T1$="F/O TRIANGOLARE (1=SI/0=NO):"
15280 PRINT T1$;(VO%(V-1,4) AND 16)/16;:
Q$=""
15290 INPUT Q$:IF Q$<>"" THEN VO%(V-1,4)
=(VO%(V-1,4) AND 238)OR(VAL(Q$)*17)
15300 REM******************************
15305 T1$="DISABILITAZIONE VOCE (1=SI/0=
NO):"
15310 PRINT T1$;(VO%(V-1,4) AND 8)/8;:Q$
=""
15320 INPUT Q$:IF Q$<>"" THEN VO%(V-1,4)
=(VO%(V-1,4) AND 246)OR(VAL(Q$)*9)
15330 REM******************************
15340 T1$="MODULAZIONE AD ANELLO "+STR$(
V)+"CON"+STR$(V-1-3*(V=1))+" 1=SI/0=NO"
15350 PRINT T1$;(VO%(V-1,4) AND 4)/4;:Q$
=""
15360 INPUT Q$:IF Q$<>"" THEN VO%(V-1,4)
=(VO%(V-1,4) AND 250)OR(VAL(Q$)*5)
15370 REM******************************
15380 T1$="SINCRONIZZAZIONE DI"+STR$(V)+
"CON"+STR$(V-1-3*(V=1))+" 1=SI/0=NO"
15390 PRINT T1$;(VO%(V-1,4) AND 2)/2;:Q$
=""
15400 INPUT Q$:IF Q$<>"" THEN VO%(V-1,4)
=(VO%(V-1,4) AND 253)OR(VAL(Q$)*2)
15410 REM******************************
15420 T1$="FASE DI ATTACCO (0-15):"
15430 PRINT T1$;(VO%(V-1,5) AND 240)/16;
:Q$=""
15440 INPUT Q$:IF Q$<>"" THEN VO%(V-1,5)
=(VO%(V-1,5) AND 15)OR(VAL(Q$)*16)
15450 REM******************************
15460 T1$="FASE DI DECADIMENTO (0-15):"
15470 PRINT T1$;VO%(V-1,5) AND 15;:Q$=""
15480 INPUT Q$:IF Q$<>"" THEN VO%(V-1,5)
=(VO%(V-1,5) AND 240)OR VAL(Q$)
15490 REM******************************
15500 T1$="FASE DEL SOSTENERE (0-15):"
15510 PRINT T1$;(VO%(V-1,6) AND 240)/16;
:Q$=""
15520 INPUT Q$:IF Q$<>"" THEN VO%(V-1,6)
=(VO%(V-1,6) AND 15)OR(VAL(Q$)*16)
15530 REM******************************
15540 T1$="FASE DI RILASCIO (0-15):"
```

```
15550 PRINT T1$;VO%(V-1,6) AND 15;:Q$=""
15560 INPUT Q$:IF Q$<>"" THEN VO%(V-1,6)
=(VO%(V-1,6) AND 240)OR VAL(Q$)
15570 REM*******************************
15580 T1$="TAGLIO FILTRO P/BASSO (0-7):"
15590 PRINT T1$;FI%(0) AND 7;:Q$=""
15600 INPUT Q$:IF Q$<>"" THEN FI%(0)=VAL
(Q$)
15610 REM*******************************
15620 T1$="TAGLIO FILTRO P/ALTO(0-255):"
15630 PRINT T1$;FI%(1);:Q$=""
15640 INPUT Q$:IF Q$<>"" THEN FI%(1)=VAL
(Q$)
15650 REM*******************************
15660 T1$="RISONANZA FILTRO (0-15):"
15670 PRINT T1$;(FI%(2) AND 240)/16;:Q$=
""
15680 INPUT Q$:IF Q$<>"" THEN FI%(2)=(FI
%(2) AND 15)OR(VAL(Q$)*16)
15690 REM*******************************
15700 T1$="FILTRAGGIO VOCE (1=SI/0=NO):"
15710 PRINT T1$;(FI%(2) AND 2↑(V-1))/2↑(
V-1);:Q$=""
15720 INPUTQ$:IF Q$<>""THEN FI%(2)=(FI%(
2) AND (255-2↑(V-1)))OR(VAL(Q$)*2↑(V-1))
15730 REM*******************************
15740 IF V<>3 THEN 15780
15750 T1$="SPEGNIMENTO VOCE 3 (1=SI/0=NO
):"
15760 PRINT T1$;(FI%(3) AND 128)/128;:Q$
=""
15770 INPUT Q$:IF Q$<>"" THEN FI%(3)=(FI
%(3) AND 127)OR(VAL(Q$)*128)
15780 REM*******************************
15790 T1$="FILTRO P/ALTO (1=SI/0=NO):"
15800 PRINT T1$;(FI%(3) AND 64)/64;:Q$="
"
15810 INPUT Q$:IF Q$<>"" THEN FI%(3)=(FI
%(3) AND 191)OR(VAL(Q$)*64)
15820 REM*******************************
15830 T1$="FILTRO P/BANDA (1=SI/0=NO):"
15840 PRINT T1$;(FI%(3) AND 32)/32;:Q$="
"
15850 INPUT Q$:IF Q$<>"" THEN FI%(3)=(FI
%(3) AND 223)OR(VAL(Q$)*32)
15860 REM*******************************
15870 T1$="FILTRO P/BASSO (1=SI/0=NO):"
15880 PRINT T1$;(FI%(3) AND 16)/16;:Q$="
"
15890 INPUT Q$:IF Q$<>"" THEN FI%(3)=(FI
```

```
%(3) AND 239)OR(VAL(Q$)*16)
15900 REM********************************
15910 T1$="REGOLAZIONE VOLUME (0-15):"
15920 PRINT T1$;FI%(3) AND 15;:Q$=""
15930 INPUT Q$:IF Q$<>"" THEN FI%(3)=FI%
(3) OR VAL(Q$)
15940 FOR I=0 TO 6:IF VO%(V-1,I)>255 THE
N GOTO 15970:NEXT
15950 FOR I=0 TO 3:IF FI%(I)>255 THEN GO
TO 15970:NEXT
15960 RETURN
15970 PRINT "SPIACENTE, MA C'E' UN ERROR
E"
15980 PRINT "RIPROGRAMMARE QUESTA VOCE."
15990 FOR I=0 TO 2000:NEXT:GOTO 15000
```

Benché questo modulo appaia lungo in modo preoccupante, è in effetti estremamente semplice. Il suo scopo è di permettere all'utente di indirizzare separatamente tutte le funzioni interessate del chip SID. I valori vengono memorizzati nelle matrici VO% e FI% finché non vengono suonati come motivo.

Commenti

Le linee 15030-15040: V viene posto uguale al numero della voce che l'utente desidera attivare. L'indirizzo 54272 è l'inizio del SID, dove le parti principali di ogni voce occupano 7 byte di dati, la posizione di partenza di ogni voce viene dunque calcolata in 15040.

Linee 15060-15130: queste due routine determinano l'ampiezza dell'impulso se l'utente desidera fare uso della forma d'onda impulsiva. Ogni valore corrente viene visualizzato e rimane immutato se si preme RETURN.

Linee 15140-15170: questa routine attiva la generazione di un rumore casuale per la voce interessata. Notate che stiamo indirizzando non già un intero byte del computer, ma un singolo bit (ci sono otto bit, o interruttori di acceso-spento, in ogni byte). A questo scopo dobbiamo fare uso delle funzioni AND e OR. Per mostrare se è attivato un dato bit, stampiamo nella matrice il valore moltiplicato logicamente (AND) con la potenza di 2 del numero del bit corrispondente — i bit sono numerati da 0 a 7 con valori crescenti da destra a sinistra. Se il bit è attivo (a 1), si ottiene lo stesso valore, altrimenti zero. Per trasformare il valore ottenuto in "zero" o in "uno", questo viene diviso per due elevato alla potenza del numero del bit. Per modificare il valore del bit desiderato, è necessario moltiplicare logicamente (AND) il valore dell'intero byte per 255 — 2↑ (numero del bit): ciò pone a zero il bit desiderato, lasciando invariati tutti i rimanenti. Il singolo bit viene ora sommato logicamente (OR) con l'uno o lo zero inseriti, così ponendo il suo valore a 1 o 0 come volevamo.

Linea 15170: notate che, sebbene il bit che desideriamo attivare sia il numero 7 (valore 128), in realtà sommiamo al byte il valore 129,

attivando così il bit 7 e il bit 0. Ciò a causa del fatto che i valori delle forme d'onda non producono in realtà alcun suono se non viene attivato il bit 0.

Linee 15180-15290: queste tre routine compiono la medesima funzione per i bit 6-4, le altre tre forme d'onda.

Linee 15300-15400: queste due routine permettono di modulare l'uscita di questa voce con la forma d'onda e l'inviluppo della nota di un'altra voce, con risultati spesso sorprendenti. Non è necessario che l'altra voce sia effettivamente predisposta per emettere il suono, ma devono essere stati introdotti i valori di forma d'onda e d'inviluppo. L'uso di queste due funzioni si può scoprire solo per tentativi.

Linee 15410-15560: queste quattro routine permettono di predisporre ATTACCO, DECADIMENTO, SUSTAIN, RILASCIO. Notate che in queste routine, invece di agire su singoli bit, si opera su gruppi di quattro. Una AND del byte con 240, seguita da una OR con un valore tra 0 e 15, agisce sui bit 0-3. Una AND con 15, seguita da una OR con un valore (0-15) * 16 agisce sui bit 4-7.

Linee 15570-15680: tramite queste tre routine si possono predisporre le frequenze a cui operano i filtri del SID. Questi valori si applicano dunque a tutte le voci per le quali sono predisposti i filtri.

Linee 15690-15890: le rimanenti sezioni permettono di attivare o no i tre tipi di filtri disponibili. Il filtro passa-alto lascia passare inalterate le frequenze superiori al valore prefissato. Il filtro passa-basso esegue la stessa funzione per le frequenze basse. Il filtro passa-banda permette di passare a una banda di frequenze centrali. Se sono attivati tutti e tre i filtri, sarà dunque ridotto il volume complessivo della nota. L'utente ha facoltà di scegliere se filtrare o no una data nota.

Linee 15730-15770: nel caso della voce 3 c'è un bit particolare che permette di escludere l'uscita della voce.

Linee 15910-15930: viene predisposto contemporaneamente per tutte le voci il volume a cui emettere le note.

Linee 15940-15990: dato che fino a questo punto non ci sono controlli d'errore sull'introduzione di valori, vengono analizzati i contenuti delle matrici, allo scopo di verificare che non vi siano valori superiori a 255: tentare di eseguire una POKE con un valore simile in un solo byte porterebbe all'arresto del programma.

Collaudo del modulo 7.1.4

Non è possibile, a questo stadio, un controllo completo di questo modulo, dal momento che non c'è ancora alcuna routine per suonare effettivamente un motivo.
È tuttavia possibile una prova di natura logica annotando accuratamente i valori introdotti e stampando di seguito i contenuti di VO% e FI% per controllare, con l'aiuto del listato, che corrispondano a

ciò che è stato introdotto. Per esempio, se è stato introdotto il valore massimo per ogni argomento, con la voce posta a 1, allora VO%(0,2-6) dovrebbe contenere 255.

MODULO 7.1.5

```
13000 REM********************************
13010 REM MATRICI MOTIVO
13020 REM********************************
13030 RESTORE:FOR I=0 TO 95:READ A:NEXT
13040 TL=0:FOR I=0 TO 2:VL=1
13050 READ NT,NL:IF NT=0 THEN 13140
13060 WW=VO%(I,4):IF NT<0 THEN NT=-NT:WW
=1
13070 IF NL<>1 THEN 13090
13080 HF%(I,VL)=NO%(0,NT):LF%(I,VL)=NO%(
1,NT):WF%(I,VL)=WW:VL=VL+1:GOTO 13050
13090 FOR J=1 TO NL-1:HF%(I,VL)=NO%(0,NT
):LF%(I,VL)=NO%(1,NT):WF%(I,VL)=WW
13100 VL=VL+1:NEXT J
13110 HF%(I,VL)=NO%(0,NT):LF%(I,VL)=NO%(
1,NT):WF%(I,VL)=WW-1:VL=VL+1
13120 IF WF%(I,VL-1)<0 THEN WF%(I,VL-1)=
1
13130 GOTO 13050
13140 IF VL>TL THEN TL=VL
13150 NEXT I
13160 RETURN
```

Questo modulo estrae la nota specificata dall'utente sotto forma di DATA e la compila in una forma che il SID possa suonare. Questa operazione è necessaria perché permette al programma di trattare note di lunghezza diversa. Le note effettivamente suonate dal programma sono tutte della stessa lunghezza, determinata da un ciclo di temporizzazione. Le note più lunghe vengono ottenute collegando una serie di note fino a raggiungere la lunghezza desiderata — non è avvertibile alcuna separazione tra le singole parti delle note. Le lunghezze possono variare per le singole voci, ma ovviamente debbono essere tali da mantenere coordinate tutte le voci che devono essere usate.

Per costruire i dati di un motivo, tutto ciò che bisogna fare è registrare, per ogni nota, il relativo numero tratto dal Manuale, e la sua lunghezza. L'unità di misura in cui viene registrata la lunghezza dipenderà dalla nota più breve che si desidera suonare. Accorciando il ciclo di temporizzazione alla linea 14180 sarà possibile suonare note più brevi, ma ciò significherà che le note più lunghe dovranno essere formate da più unità elementari. In questo caso l'inconveniente è che ogni singola unità di una nota, indipendentemente dal tempo determinato dal ciclo di temporizzazione, richiederà uno spazio in ciascuna delle matrici LF%, HF% e WF%, cosicché, quanto più si abbrevia il ciclo di temporizzazione, tanto maggiore sarà la memoria necessaria per contenere un motivo di una data lunghezza.

Commenti

Linea 13030: dal momento che le predisposizioni delle voci e i valori delle note vengono modificati durante lo sviluppo del pezzo, sarà necessario leggere più volte i dati. Poiché la tabella dei valori delle note è posta prima dei dati relativi al pezzo, è necessario azzerare il puntatore alle DATA tramite RESTORE e rileggere ogni volta la tabella delle note fino all'inizio dei dati del motivo da suonare. Il puntatore non si può posizionare in un qualsiasi punto del programma, a piacere; si può solo riportare all'inizio dei dati o lasciare dove si trova, e puntare alla linea di dati seguente l'ultima letta.

Linea 13040: questo ciclo assicura che vengano compilati i dati del motivo per ogni voce. La variabile VL viene utilizzata per memorizzare la lunghezza del motivo per ogni singola voce.

Linea 13050: vengono letti dai dati del motivo il valore e la lunghezza della nota. Se il valore è zero, il programma considera completi i dati della voce e si porta alla successiva.

Linea 13060: si legge dalla matrice VO% il valore della forma d'onda della voce corrente. Se il valore della nota è un numero negativo, il valore della forma d'onda viene ridotto di uno, azzerando così il valore del bit zero relativo alla forma d'onda, e questo produrrà un silenzio di lunghezza NL, invece di un suono.

Linee 13070-13080: se la lunghezza della nota è 1, i dati della frequenza alta e della forma d'onda vengono memorizzati nella matrice corrispondente e la lunghezza della voce viene aumentata di uno.

Linee 13090-13130: se la lunghezza della nota è maggiore di 1, vengono riempite NL — 1 posizioni successive nella matrice con i valori di frequenza e forma d'onda. All'ultimo elemento della nota, il valore della forma d'onda viene ridotto di 1, permettendo così alla nota di affievolirsi in modo naturale. Se per una voce non vengono inseriti dati diversi da 0,0 (cioè non desiderate utilizzare quella particolare voce), è possibile che la forma d'onda assuma il valore di —1, cosa che fermerebbe il programma se si tentasse di eseguire una POKE. La linea 13120 controlla che ciò non accada.

Linea 13140: la lunghezza del motivo per la voce corrente (VL) viene confrontata con TL, che contiene inizialmente zero, e, in seguito, la lunghezza della più lunga voce del motivo. Questo assicura che, durante l'esecuzione, non ci si fermi prima dell'esaurimento di tutto il contenuto di ogni voce.

Collaudo del modulo 7.1.5

Ancora una volta non è possibile collaudare completamente il modulo finché non venga effettivamente suonato il pezzo. Tuttavia, se vengono inseriti dei dati analoghi a quelli del modulo 8, con una forma d'onda predisposta almeno per la voce 1, chiamando questo modulo si dovrebbero porre i corretti valori alti e bassi delle frequenze in HF% e LF% e quelli relativi alle forme d'onda in WF%.

MODULO 7.1.6

```
14000 REM#********************************
14010 REM SUONA BRANO
14020 REM*********************************
14030 FOR I=54272 TO 54296:POKEI,0:NEXT
14040 FOR I=0 TO 2:VS=54272+7*I
14050 POKE VS+2,VO%(I,2)
14060 POKE VS+3,VO%(I,3)
14070 POKE VS+5,VO%(I,5)
14080 POKE VS+6,VO%(I,6)
14090 NEXT I
14100 POKE 54293,FI%(0)
14110 POKE 54294,FI%(1)
14120 POKE 54295,FI%(2)
14130 POKE 54296,FI%(3)
14140 FOR I=1 TO TL
14150 POKE54272,LF%(0,I):POKE54279,LF%(1
,I):POKE54286,LF%(2,I)
14160 POKE54273,HF%(0,I):POKE54280,HF%(1
,I):POKE54287,HF%(2,I)
14170 POKE54276,WF%(0,I):POKE54283,WF%(1
,I):POKE54290,WF%(2,I)
14180 FOR TT=1 TO 80:NEXT TT,I
14190 FOR TT=1 TO 200:NEXT:POKE54296,0
14200 POKE54276,1:POKE 54283,1:POKE 5429
0,1
14210 RETURN
```

Questo modulo introduce nel SID le caratteristiche delle voci predisposte nel modulo 4, di seguito introduce i valori delle frequenze, insieme con la forma d'onda desiderata, per produrre le note che costituiscono il motivo.

Commenti

Linea 14030: il chip SID viene inizializzato forzando a zero ogni sua locazione.

Linee 14040-14090: vengono forzate nel SID le predisposizioni della voce specificate al modulo 4.

Linee 14100-14130: le predisposizioni relative ai filtri sono comuni a tutte le voci, vengono così introdotte una volta sola.

Linee 14140-14180: fino al raggiungimento della lunghezza del motivo (TL), per ogni nota, vengono posti nei primi due byte di ogni locazione di voce del SID i valori per la frequenza alta e bassa, seguiti, nella quinta locazione, dalla forma d'onda. Questa operazione attiva la nota desiderata, che viene suonata per la durata del ciclo alla linea 14180.

Linea 14190: alla conclusione del brano si utilizza un ciclo lievemente più lungo, per fare sì che il suono si estingua; le tre forme d'onda vengono quindi poste sul silenzio.

Collaudo del modulo 7.1.6

Piuttosto semplice. Se avete introdotto dei dati, inizializzato il programma, attivato almeno una voce e compilato il brano, dovreste ora poter suonare la vostra opera. I dati d'esempio forniti al Modulo 8 suonano una scala di DO con le prime due voci.

MODULO 7.1.7

```
17000 REM#********************************
17010 REM FILE DATI
17020 REM*********************************
17030 INPUT "▓▓POSIZIONARE IL NASTRO, QU
INDI PREMERE   ▓RETURN▓--";Q$
17040 PRINT "▓▓1)SALVATAGGIO DATI":PRINT
 "2)CARICAMENTO DATI"
17050 INPUT "▓FUNZIONE RICHIESTA:";Q
17060 ON Q GOTO 17070,17130:RETURN
17070 OPEN 1,1,2,"MUSICA":PRINT#1,TL
17080 FOR I=0 TO 1:FOR J=0 TO 95:PRINT#1
,NO%(I,J):NEXT J,I
17090 FOR I=0 TO 2:FOR J=0 TO 6:PRINT#1,
VO%(I,J):NEXT J,I
17100 FOR I=0 TO 3:PRINT#1,FI%(I):NEXT
17110 FOR I=0 TO 2:FORJ=0TOTL:PRINT#1,LF
%(I,J);R$;HF%(I,J);R$;WF%(I,J):NEXT J,I
17120 CLOSE1:RETURN
17130 OPEN 1,1,0,"MUSICA":INPUT#1,TL
17140 FOR I=0 TO 1:FOR J=0 TO 95:INPUT#1
,NO%(I,J):NEXT J,I
17150 FOR I=0 TO 2:FOR J=0 TO 6:INPUT#1,
VO%(I,J):NEXT J,I
17160 FOR I=0 TO 3:INPUT#1,FI%(I):NEXT
17170 FOR I=0 TO 2:FORJ=0TOTL:INPUT#1,LF
%(I,J);HF%(I,J);WF%(I,J):NEXT J,I
17180 CLOSE1:RETURN
```

Si tratta di un modulo standard di file dei dati, che permette di salvare su nastro il brano introdotto.

MODULO 7.1.8

```
19000 REM#********************************
19010 REM DATI VOCE 1
19020 REM*********************************
19030 DATA 34,2,34,2,39,2,41,2,43,2,39,1
,38,1,36,2,48,1,48,1,0,0
20000 REM#********************************
20010 REM DATI VOCE 2
20020 REM*********************************
20030 DATA 36,2,38,2,40,2,41,2,43,2,45,2
,47,2,48,2,0,0
21000 REM#********************************
21010 REM DATI VOCE 3
21020 REM*********************************
21030 DATA 0,0
```

Commenti Linee 19000-21030: dati di esempio per l'esecuzione di una scala di DO. Notate che, se desiderate usare una voce, dovete ancora introdurre per questa 0,0.

Riepilogo Questo programma sarà solo l'inizio delle vostre avventure con le possibilità sonore del 64. È un cavallo di battaglia che vi permetterà di sviluppare la vostra musica, per trasferirla poi ad altri programmi, usando solo le matrici in cui è memorizzata e il modulo 6, che effettivamente la suona.

Ulteriori sviluppi Posto di non incorrere in troppe limitazioni di memoria, ci sono svariate direzioni in cui è possibile estendere il programma:
1) Perché non concedersi la possibilità di inserire la forma d'onda per ogni singola nota, invece che soltanto per ogni voce? Bisogna unicamente aggiungere un terzo valore per ogni nota da suonare.
2) Volendo suonare brani più lunghi, perché non adattare il programma in modo da utilizzare un ciclo di lunghezza variabile per imporre la lunghezza di ogni nota? Si risparmierebbe in questo modo notevolmente sulla quantità di spazio necessario nelle matrici. Nella matrice finale, della nota dovrebbero essere memorizzati solo il valore e la lunghezza, che è utilizzata per determinare la durata del ciclo di temporizzazione. Sfortunatamente tutto ciò si può fare solo per una voce, dato che la temporizzazione impone la durata della nota per tutte e tre le voci.
3) Volendo brani più lunghi con più di una voce, perché non dare al programma la possibilità di compilare parti di un brano? Queste verrebbero raccolte in seguito da parte di un programma esecutore che non necessita di enunciati DATA, di notevole spreco in termini di memoria.

David Lawrence ha pubblicato numerosi libri sui computer e sulle tecniche di programmazione. Collabora a importanti periodici e pubblicazioni di divulgazione scientifica.

Questo non è soltanto un libro da leggere, un volume cioè da considerare come un mezzo d'apprendimento di nuove tecniche di programmazione, è anche uno strumento di lavoro, una vera e propria collezione di programmi per il popolare home computer Commodore 64, scelti e ordinati per utilizzare al meglio le particolari caratteristiche della macchina.

Tutti provati autonomamente, i programmi offrono campi di applicazioni che avrebbero potuto aprirsi soltanto a chi fosse disposto a comperare costosi software in commercio oppure già in grado di scrivere complessi programmi per soddisfare le proprie esigenze.

Sono scritti in forma 'modulare'. Ciò significa che sono costituiti da unità funzionali, chiaramente identificabili che, una volta comprese, si possono estrarre e reimpiegare secondo necessità.
Ogni modulo, dove si riferisca a qualcosa di nuovo, viene commentato e si forniscono istruzioni per provare i programmi a mano a mano che i moduli vengono inseriti. L'approccio modulare aiuta a evitare che i programmi diventino inestricabili grovigli di errori.

Per la chiarezza dei temi esposti e per l'immediatezza d'uso, il libro risponde a una domanda molto diffusa di strumenti per programmare con facilità, in particolare è rivolto agli utenti del Commodore 64, ma anche, in generale, ai possessori degli altri home computer o elaboratori personali di uso domestico.

Un volume quindi di pratico impiego e un'occasione di approfondimento delle tecniche di programmazione.

0025633–9

Lire **18.000**
(IVA inclusa)

LAWRENCE

IL MEGLIO DAL COMMODORE 64